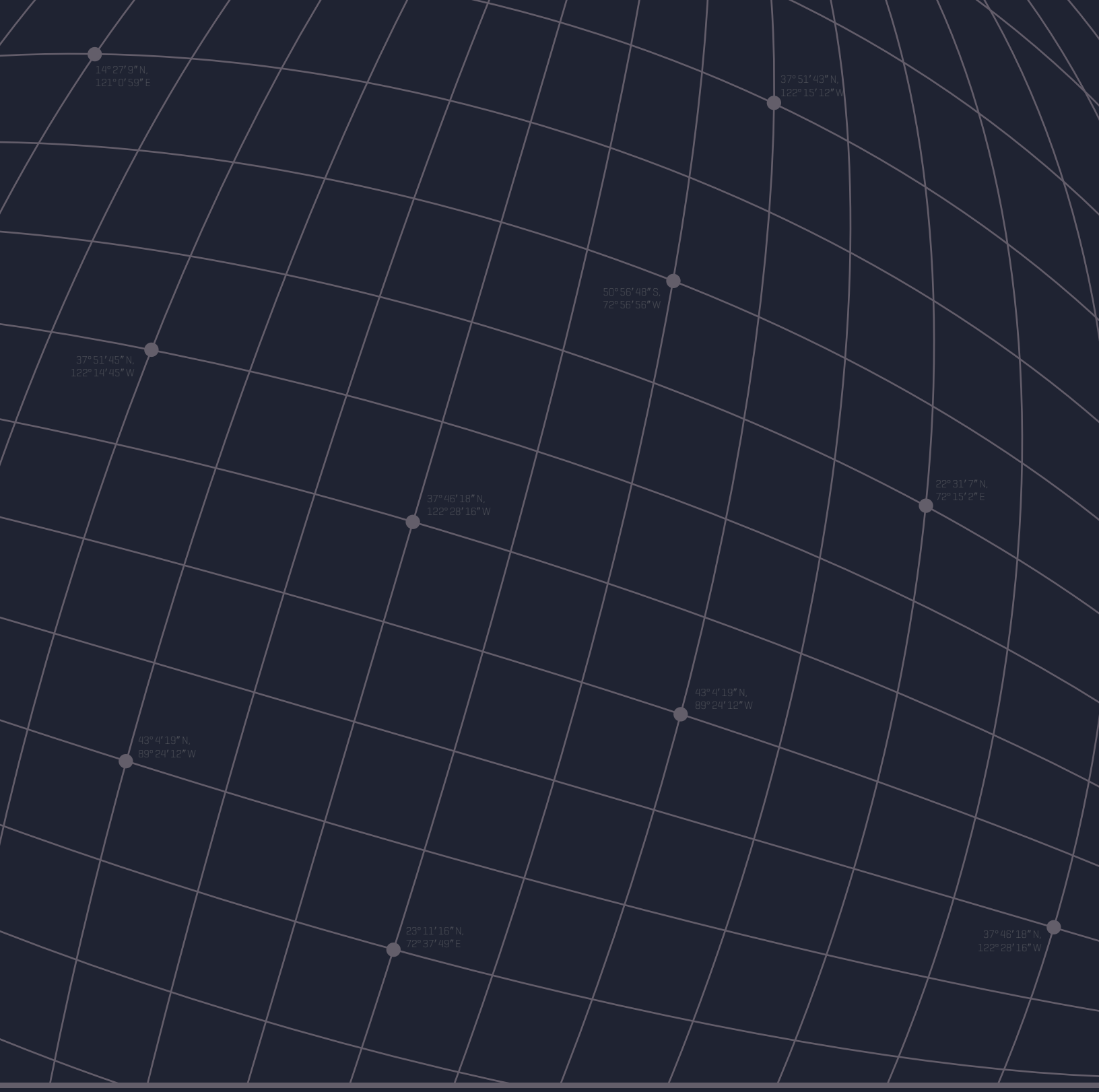
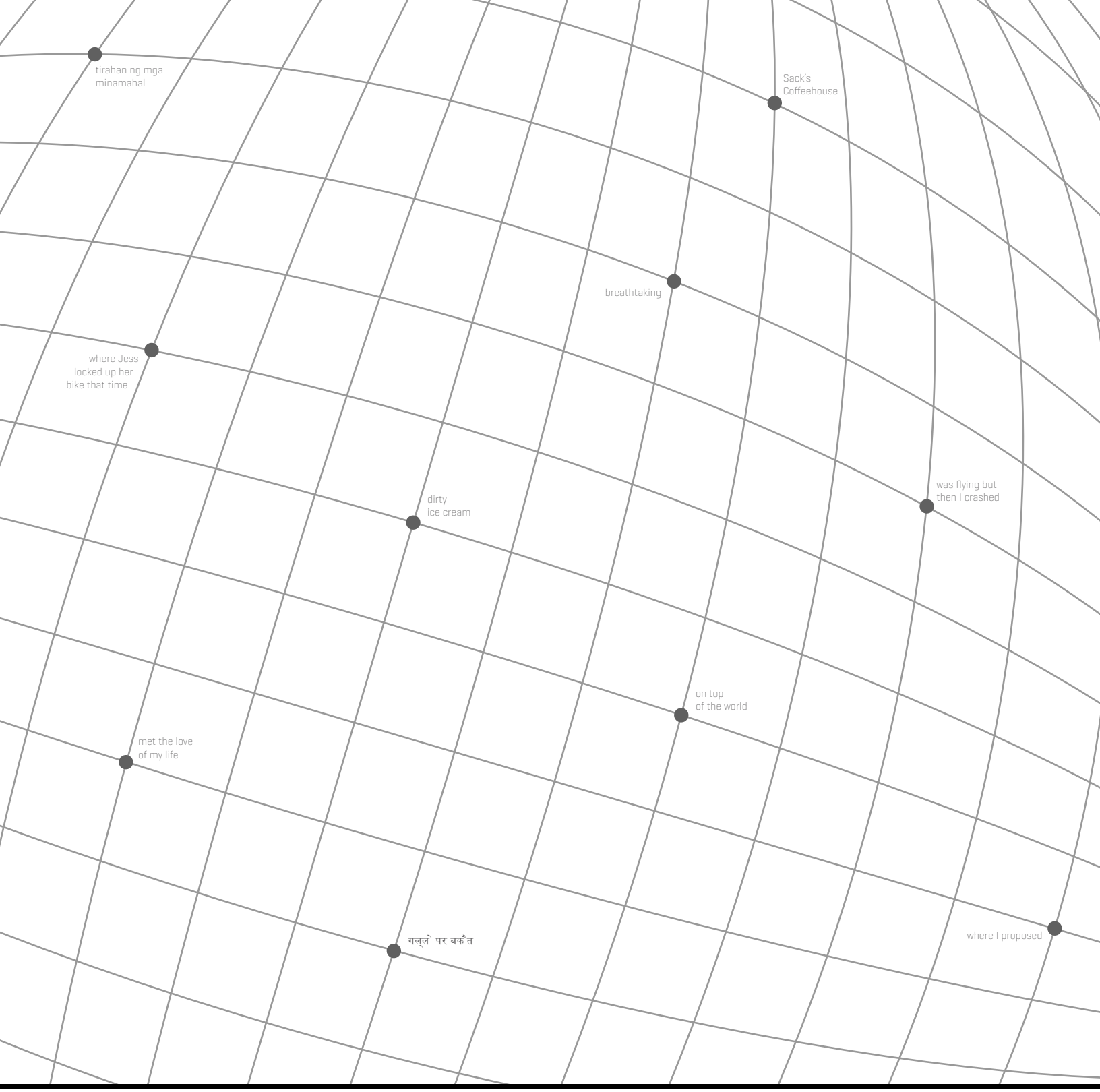



# MeLo

## The Meaningful Location Project

Nick Doty
Ryan Greenberg
Mohit Gupta
Karen Nomorosa
Stephanie Pakrul
Advised by Deirdre Mulligan




# MeLo

## The Meaningful Location Project

Nick Doty
Ryan Greenberg
Mohit Gupta
Karen Nomorosa
Stephanie Pakrul
Advised by Deirdre Mulligan

# Contents

# Introduction

Increasingly, people's devices know where they are. Smart phones combine triangulation of cell phone towers with the global positioning system (GPS) to determine their position within tens of meters. Laptop computers can guess where they are based on the WiFi networks they can see. This process, where a device determines its location, is called geolocation. At the same time there has been an increase in the number of services that consume location data: social network sites let users share where they are, location brokers provide a centralized mechanism for using location data, and even services that were not previously location-oriented have augmented their offerings to make use of place.

Location frames people's experience of the world: what is possible, what is interesting, and what is relevant depend in part on where a person is. Widespread adoption of location-aware devices facilitates exciting new services and new ways to augment existing services. Search results can be personalized based on the place you are now, giving you movie times for the nearest theater or the weather forecast for here. More location-aware devices makes it easier to add location metadata to files, which in turn makes it possible to search digital objects based on the physical world. You could search for all the photos you took at a particular point, or the files you worked on at that place. A location-aware phone could help you meet up with someone in a crowded space, or let that person know you'll be late because you're still in traffic.

Throughout 2010, which some have termed "the year of location," many location-based services and application programming interfaces (APIs) have been released or updated (Malik 2010). Most of these services focus on one of two areas: collecting and exposing vast amounts of geographic data, and allowing users to share their current location with others. Some services in the first category are GeoAPI, SimpleGeo, and Yelp, which are focused on aggregating location information that can be used by applications. These services are useful for getting information about new places, or supporting new navigations through space. For developers, they offer data and services to query and annotate the real world with additional information. Services like Foursquare and Gowalla currently emphasize the user's current location and the social graph.

The increased proliferation of location data is accompanied by justified concerns. Privacy and security are always an issue when a service is collecting, transmitting, or storing users' personal information. This is heightened when dealing with location data because of special concerns that people have about their location. Knowing a person's location gives someone the ability to find that person. Access to a person's location history can reveal where a person lives, works, and shops, as well as more intimate details including schedule, social behavior, political and religious beliefs, and sexual orientation.

We believe that location-based services in both of these areas can benefit from a system that takes into consideration the complex social nature of location. Such a system could also serve to increase people's privacy with respect to geolocation data.

Here we describe the structure of such a system and an implementation we developed. While there are some limits to present-day use of geolocation, we developed our system premised on the belief that even more devices in the future will be capable of providing near real-time information about their position.

## The Problem with Location

At some level all location-based devices and services rely on the universal system for describing positions on Earth, latitude and longitude. Any point on the Earth's surface can be represented by a latitude and longitude coordinate pair. These points can be used to construct other geographic primitives like bounding boxes and polygons that describe different parts of the globe (see figure 1). Despite their ubiquity, coordinates have the significant drawback of not being readily intelligible to people. Without special knowledge, people have no idea what places are represented by coordinate pairs. As a result, translation between coordinates and names that people understand is a fundamental issue in dealing with geographic data.

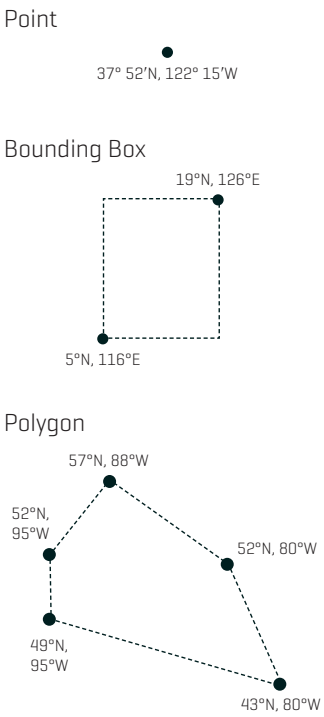



*Figure 1.* Latitude/longitude coordinate pairs describe points on the Earth's surface. These points can be used to build other geographic primitives. A bounding box is the rectangle formed by the two points of opposing corners. The shape of complex geographic entities can be described by a polygon made of an arbitrary number of points.

Translating names for places to the corresponding coordinates is called geocoding. Mapping services like those provided by Yahoo! Maps and MapQuest rely on geocoding to provide directions. When a user types an address like "1517 Shattuck Ave, Berkeley, CA" a provider geocodes this address to get its coordinates. Conversely, reverse geocoding is the process of translating geographic coordinates to names that have more meaning to people. A set of coordinates can have descriptions at different levels, so there are many kinds of reverse geocoders that produce different outputs. Given a set of coordinates, a reverse geocoder could return a street address, the political geographic hierarchy of a place, or the name of a point of interest at that location.

In many contexts a street address, city name, or point of interest is a better way to refer to a coordinate pair. Think about the question, "Where are you?" You can imagine situations where people would respond by saying "I'm at Second and Jackson," or "I'm in Oakland," or "I'm at Starbucks." These options, however, cover just a subset of the ways people answer this question as existing reverse geocoders are limited in the scope of answers they can provide. Reverse geocoders that provide canonical place names like "Peet's Coffee" or "Subway" generally have databases that only list commercial venues and notable landmarks. They are also fairly static and lack information about transient activities or events. They do not provide any means for recording common, personal names for places, such as "at work," "Dad's house," "home," or "at the concert".

## The Meaningful Location Project (MeLo)

In order to provide rich, personal, location-enhanced services, we have built a system that performs a new type of reverse geocoding that supports more meaningful location names. This service is not merely a location application, but rather a platform that facilitates a new class of cross-functional applications to go beyond the limited scope of current location-based services. Instead of translating a coordinate pair to the name of a city, our reverse geocoder combines coordinates with a user's identity to offer a name like "home" or "favorite bakery." Our system relies on annotations from users and their historical location data to perform this translation.

Annotations are a person's description of where he or she has spent time. They are a person's description of a place: personal names for it, adjectives that describe it, activities done there, and categories that a place belongs to. Coupled with historical knowledge of a user's location, a system can translate between coordinates and this personal vocabulary. Additionally, these location points can then be supported without their original indexical references.

In order to do this, a user authenticates a location broker (such as Fire Eagle) with MeLo, which will import their location in near real time and share the users' full location history. The user can then use the MeLo interfaces and backend tools to easily annotate ("tag" or "categorize") their location data, making it more personally meaningful.

Data in the system is exposed through a REST API that developers can use to offer additional services to users. Tools built using the API can not only allow the user to use their abstract conceptions of place, but also provide additional views and analysis of the data. This API-centric approach allows developers to create additional applications as test cases for how users engage with features provided by the API.

Various visualizations created by the team leverage these annotated location histories to present users with information that enables reflection, search, and analysis of location data. Additionally, the system allows the user to then share their location data in ways that preserve their privacy.

In the following sections we describe the research and user testing that informed the development of our system, and the methodology used to construct it. We give an overview of the system's architecture including conceptual models, data structure, and the API that provides access to the platform. Finally we conclude by describing several proof of concept applications that we built to showcase the potential of this platform and discussing the implications in related fields.

# Methodology And Tools

In keeping with the inherently interdisciplinary approach of the School of Information, we sought to explore the concept of meaningful location through an integration of research and software development. Traditional research, whether litera-

ture review or user studies, both informed how we built our software (determining specifications, prioritizing features, changing functionality) and was enabled by the concrete existence of software (allowing for functional prototypes and real-world extended testing). Though the end result of MeLo is primarily a software product, we reject the notion of a bright line between development and research, or between purely professional and purely academic work; we believe good software is produced by understanding and exploring a problem in depth.

## Research Methodology

People have always shared their location with others both explicitly in conversations and implicitly when describing their behaviors and routines to others. Given this long history, we wanted to improve our understanding of place and location histories in a broad sense, before finalizing features and implementation details.

Technical infrastructure to support location is often designed primarily from a computing systems perspective. The rise of consumer products and Internet-based technologies for geolocation have exposed the field of "neogeography": a light-weight approach to mapping and traditional geography that focuses on simple tools (like the Google Maps API) and personal data like geo-tagged photos (Turner 2006). This light-weight approach and democratization of tools has allowed for a boom in location-based service development. But this has also introduced a reliance on location data based on coordinates which provide an easy interchange format (see *The Problem with Location*, above).

User experience research provides for streamlining the features and designs of software products, and recently we have seen the success of applications designed with the users' motivation in mind, like the "check-in" services Foursquare and Gowalla (Gale 2010, Yahoo! Campus Sunnyvale). In the design of MeLo we relied both on technical discussions about location-enhanced services and often-overlooked theoretical material from philosophy, critical geography, urban studies, and user experience research for context-aware systems.

### *Theory*

The use of location to provide enhanced services to users is often modeled as part of context (Dey et al. 2001). The debate around context-aware computing—also referred to as pervasive computing (Ark and Selker 1999), embodied interaction (Dourish 2001) and ubiquitous computing (Weiser 1991)—can be useful to describe the design and research methodology for MeLo. While Dey et al. have argued for context as "any information that is relevant to the interaction between a user and an application," Dourish calls for a shift in methodology. Dourish draws from literature in phenomenology that "turn analytic attention away from the idea of a stable external world that is unproblematically recognized by all, towards the idea that the world, as

we perceive it, is essentially a consensus of interpretation." Others have argued that a phenomenological approach moves focus from the system as an artifact for "sensing" information, to an interactive system that has a physical interface (Svanæs 2001). This allows for a methodology where the technical system is always evaluated with a focus on the interactions offered.

A similar recognition of the diverse consensus of interpretation is present in the literature of humanistic geography. Yi-Fu Tuan is credited with initiating the (sub-)field in the 1960s and 70s, which recognized a distinction between "space"—the objective features of a location—and "place"—the interpreted, experienced meanings of a space to its inhabitants (Daniels 1992; Tuan 1977 ). Later, critical humanistic geography underlined the diversity of views of place:

> "Places [...] are not so much bounded areas as open and porous networks of social relations. [...] identities will be multiple [...] And this in turn implies that what is to be the dominant image of any place will be a matter of contestation and will change over time" (Massey 1994).

We see that same diversity of ontology (what the important types and relationships are) in modern philosophy, particularly Quine's theory of ontological relativity (Quine 1951).

The literature of geography and architecture also emphasizes the importance of duration, motion, or activity to a nuanced understanding of location: from Baudelaire's notion flâneur who understands the city by walking through it and de Certeau's idea that walking through the city produces the meaning of the city itself. We believe that location histories (rather than single points or check-ins) provide a richer description of location. Other more technical approaches to managing location histories have looked at using statistical modeling to identify various higher level entities (such as places, paths and destinations) from raw GPS data (Eagle and Pentland 2009; Hariharan and Toyama 2004; Smith 2007). Through the development process we have looked at using these algorithmic developments to realize the understandings of place developed in geography and architecture.

MeLo isn't the first to suggest that we recognize the different ways to see a place or the diverse views that people may have of the same place; for example, Alistair Edwardes suggested the use of personalized names in location-based services (Edwardes 2009) and CMU posited systems for personalized naming of locations based on their empirical study of real-world naming (Zhou et al. 2005; Lin et al. 2009). We see the same theoretical notions in our everyday understandings of how people refer to location. When asked, "Where are you?" people very often respond with non-geographical terms ("I'm on the bus" or "I'm at home") and different people often describe the same "place" in very different ways. To give one example that's close to home: Sack's Coffeehouse might be a place of solemn, anti-social productivity for one grad student, a mundane place of business for a barista, and a loud, social meeting place for a Rockridge mother. To understand Sack's, you must understand this multiplicitous, contested view of the place, not just the objective characteristics of its geolocation and the fact that it sells food and drinks. As a result, MeLo was designed and built to accept distinct personal annotations of the same geographic locations.

At the same time, information systems commonly benefit in a pragmatic way from a consistent, unified ontology wherever possible: agreement on a particular ontology or controlled vocabulary is what makes it useful. This point is made explicitly by Barry Smith in explaining the different roles ontology plays in philosophy and information science (Smith 2003). We want to take advantage of that usefulness in MeLo, so the system is designed, where possible, to understand that "South Hall" is a single name that many people share for the same building and that "Home" and "Work" are common designations for places that have special meanings for people. A common hierarchy for categories of places can put less of a burden on users to define their own hierarchies and can allow for intelligent, pre-created visualizations and rules for sharing.

### *Privacy*

Privacy of location data is an important consideration that is expressed by users and is extensively discussed in literature (Snekkenes 2001). During the development of MeLo, we aimed to provide our test users with precise and clear information about how we would use their data. We focused on understanding privacy issues around location from the perspectives of cryptography, policy and user research. Our efforts relating to privacy were not limited only to sharing information between users but also towards protection against subpoenas and data breaches of the MeLo system itself.

Apart from conducting user studies for the platform and applications we designed, we looked at various studies that explore how users conceptualize location disclosure. In particular, we were concerned with how people think about revealing their location to others and what kinds of activities usually require sharing of location information. Tsai et al. documented existing location-sharing services and the varying privacy controls they provide for users (Tsai et al. 2010). Studies by Consolvo et al. have talked about the various issues people think about when sharing location information. They identify that questions about "*who* is requesting, *why* do they need to know, *what* would be most useful to them, and am I *willing* to share that" location information are all essential towards making a decision concerning location information (Consolvo et al. 2005). Though we allow the user to make these decisions when sharing location, other readings and our own studies raise other questions. These questions such as how to design an interface for the user to express this information and how to handle privacy and sharing that can not be expressed discreetly in terms of person to person sharing are discussed later in the report.

Our concern for an understanding of location in everyday contexts is expressed in a study about location sharing practices (Bentley and Metcalf 2008). They recorded everyday phone conversations between people to learn about how "location and activity disclosures fit into everyday communication, awareness and planning." Similar studies have highlighted the extensive use of location abstractions that are shared between people (e.g. "home," "school," "where we met last time").

In many popular applications (Gowalla, Foursquare, Brightkite), privacy in location-sharing is achieved in large part by

users' explicitly controlling when they document their location (preventing accidental disclosures), which is considered one of the advantages of the "check-in" model (Kirkpatrick 2010). We believe that by using personal categories to granularly disclose categories of location, users will feel more comfortable sharing location without having checked in explicitly.

## User testing methodology

In the design process for MeLo we did not start with an initial, comprehensive needs-finding phase that defined the final architecture. Instead we progressively built the system and engaged with users to understand their interpretations of its value. This approach of evaluating tools not only in the context of need or ease of use follows from discussions of tools (Merleau-Ponty 1962). The body has an ability to adapt an extend itself through external devices, that change perceptions and at the same time become a part of it.

When we first started talking to users about the concept of MeLo, we wanted to understand what people would want to share about location information and why. Our initial user interviews were to validate and narrow our models of location. We asked questions like, "Other than home and work [or school], what are the most important locations in your life?". These questions were intended to not only get at literal responses, but also understand how people spoke about location—what words did they use, what locations did they emphasize? Our initial contextual inquiry was in conjunction with the development of two related mobile applications—the mobile tagging interface and a rules-based location sharing application. This inquiry was conducted with eight users in our target market, technically savvy users in their 20s–30s. Some of these people currently used location-based services such as Foursquare, but all were at least familiar with mobile devices and information-sharing services like Facebook and Twitter. We asked these individuals to tell us about their current practices with mobile devices and location services. They demonstrated the situations in which they use these applications, and what they did and did not consider them appropriate for. We also looked for ways that they accomplish tasks now, like remembering a certain location for later, or using the privacy controls of a system.

Throughout the development process, we evaluated our designs based on the uses that people expected of the system, with a focus towards recognizing the ecosystem in which MeLo exists. MeLo, being a platform, had to be generalizable to be applicable for various uses, which is sometimes looked down upon by subjective approaches to design. This is because Dourish and others have argued that context "is an occasioned property, relevant to particular settings, particular instances of action and particular parties to that action" (Dourish 2003). Therefore, we started with very loose conceptions of place, and incrementally designed features that were relevant to users' interactions with the system. Our focus on designing visualizations for users as opposed to statistical modeling of location histories, draws from our goal of engaging with user perspectives to define the purposes

of the system.

As a result, completing a usable alpha version of the MeLo website and tagging interface was a top priority in order to be able to get feedback once users could actually try out the system for some period of time. We also knew it wouldn't take many users to unveil most critical usability issues (Nielsen 2000), and wanted to get to these quickly. We used the initial project presentation as a recruitment tool for testing out MeLo and six people signed up to be early testers. Once a functional version was complete, we did in-depth interviews with three users where they showed us how they understood concepts in the interface and how they used the application. Members of the MeLo team were also able to start using the system at this stage, connecting our own variety of devices. Simultaneously, we created low-fidelity prototypes of the mobile applications. This was tested with six users, using a talk-aloud approach where the user spoke about their thoughts while using the interface, which was recorded on video. Even though this prototype testing was focused on the mobile applications, several issues came up with general MeLo concepts, or cross-platform issues that also existed in the web interface. For both the mobile and web applications, the location tagging interface was a strong area of focus for iteration. We then did another round of brainstorming, and independently sketched potential interfaces in parallel before coming together to refine a new interface design.

Not all our testing focused on end users; we believe that the design of APIs and platforms can gain equally from participatory design, as has been seen in the design of user interfaces. As a platform, developers are an important user base. We held a one-day "hackathon" to provide potential MeLo developers with the basic skills and access needed to create a satellite application using the MeLo API, which was attended by ten people. Our interactions with visiting developers allowed us to further refine our designs and this also served as additional quality assurance testing for the API. Although the lack of literature that explores design of programming languages and platforms as an interaction design problem points to a lack of investment in this area, we believe there will be large benefits from having a well-designed and usable API.

## Software Development Methodology

We developed the MeLo platform using an agile process so we could make rapid changes to our implementation as we learned more about the appropriate models and received user feedback. As a whole, the system went through three major cycles. Third-party applications built to use the API followed their own development schedule, as discussed in the *MeLo Ecosystem* section of the paper. Because no testing could be done on the system until location history could be collected, the first major cycle focused on ensuring that importers could communicate and exchange information with external location services. A basic tagging interface was also constructed to allow users to tag their location points once imported. The second major cycle of the product focused on refining the logic for grouping multiple points at roughly the same time and location together into 'stays'. At

this point, the design of third-party applications was started, and general information requirements were assessed to inform the design of the API architecture. Tag suggestion logic and a tagging model were also refined. Based on feedback from the location sharing team, a privacy and sharing framework was designed and exposed through the API. The third cycle incorporated all of the feedback from users. An improved tagging interface was designed and implemented based on the new tagging model, new location model, and new logic for suggesting tags. New interfaces for searching and filtering, based on user feedback, were also designed and implemented. Feedback from developers during the hackathon was used to refine and improve API methods.

## Tools

We developed the MeLo platform in Python, using Google App Engine. Before the start of our project, no team members were experts with the Python language, though most had used it before. Only one had expertise with App Engine. Because our software runs on Google's App Engine servers, we aren't responsible for maintaining servers, upgrading or troubleshooting individual machines, or expanding our infrastructure; instead, we pay Google at inexpensive, utility-level rates when usage goes above a certain level. By using App Engine we gained programmatic access to a set of services that typically require significant configuration to set up and maintain. These include data storage in BigTable, account management using Google accounts, sending email, and performance caching using memcached.

For storage, App Engine relies on Google's BigTable, one of a growing number of increasingly popular non-relational (or "NoSQL") databases that store data in key-value pairs rather than tables with a specific set of columns. As an object-oriented database model, BigTable provided some early advantages in initial ease of use (entities are straightforwardly defined as classes) but also its share of growing pains because we were all familiar with traditional relational databases with normalized structure. Because it lacks SQL's concept of joins, combining tables of data in BigTable to respond to queries is difficult or inefficient. Instead, the NoSQL approach favors pre-calculation of data, de-normalization of data models and configuration of indices: more processing is done at write-time and more copies of data are written into the database in order to speed up reading data in response to user queries. The non-relational database also has the theoretical advantage of scaling to handling large amounts of data: storing more data or handling more queries requires only adding the commodities of additional storage or processing power rather than complex strategies for dividing work between master and slave servers or techniques for sharding data appropriately.

On the front end, we used jQuery, HTML5 and CSS for quickly creating interactive web-based front-ends for our server side data and API. Using jQuery lets us quickly write JavaScript code that will function the same way on different web browsers, dramatically decreasing both the time spent developing and the time spent testing each page with multiple browsers. Rapid development of front-end interfaces allowed for quick iterations of the MeLo website and tools, allowing us to try multiple

interactive prototypes ourselves or with testers before investing large amounts of developer time. Members of our team were proficient with jQuery-focused web development before starting the project.

Version control and issue tracking with Trac allowed for versioned software development. A wiki was used for maintaining API documentation and Google Docs for collaborative writing and research.

## The Melo Ecosystem

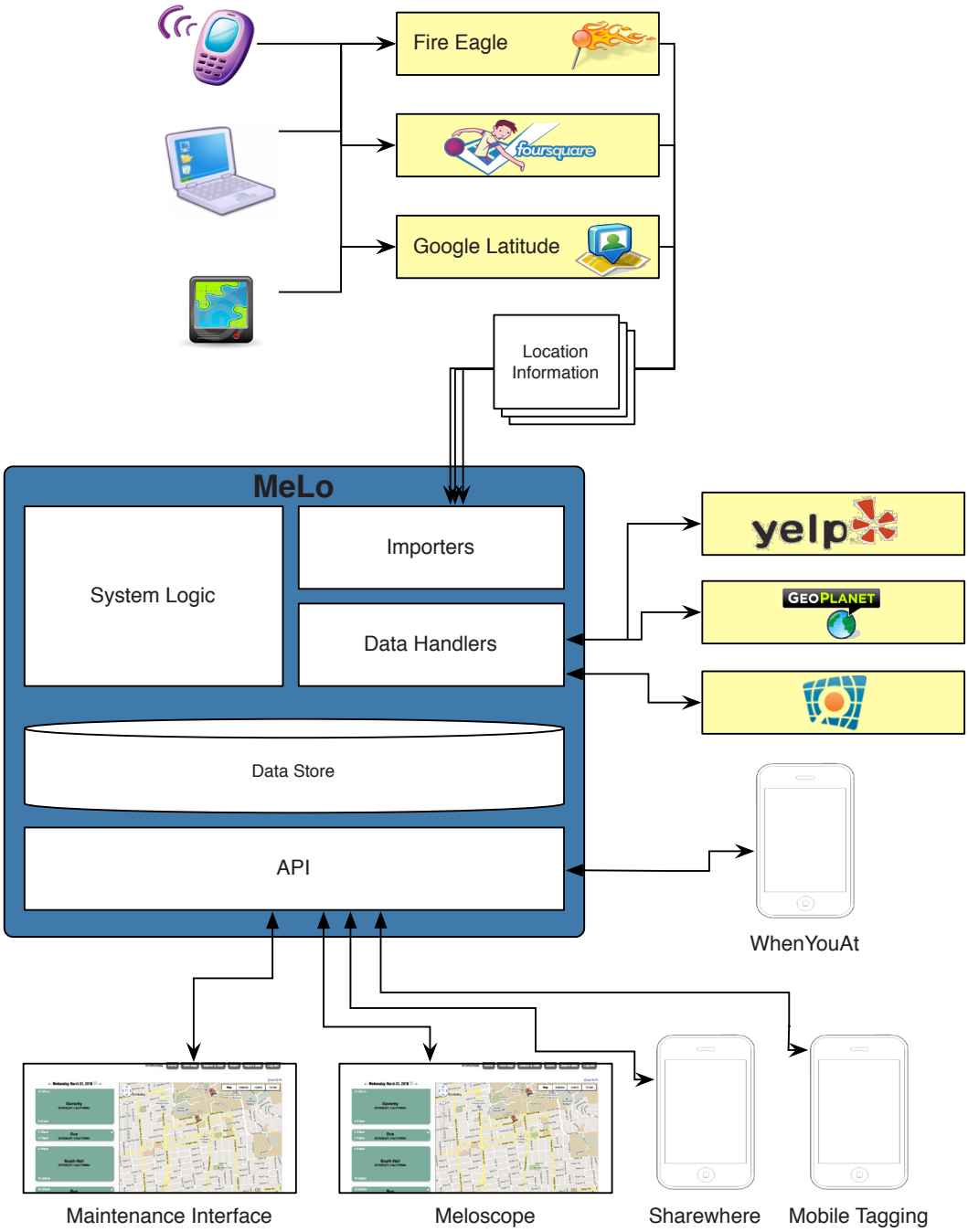



*Figure 2.* An overview of the MeLo ecosystem.

While the central focus of the MeLo project is on the platform and supporting API, it interacts with several third-party services and enables a number of third-party applications developed by teams led by MeLo project members. These external applications are composed of location-based services that provide the users' location information, as well as standard geocoders and reverse geocoders that translate between coordinates and geographic, categorical and venue name information. Developed third-party applications include meloscope, Sharewhere, and Whenyouat, which explores different uses for location information.

# The MeLo Platform

## Web and Mobile Tagging Interfaces

Users' annotations for locations are the core of the MeLo platform, which in turn makes the interface where users name and categorize location a fundamental part of the system. We refined the interface across several iterations as we developed a better understanding of the underlying location and tagging models based on user feedback. The first version of the interface simply displayed a user's location history as individual points and allowed the user to add tags to these points. Later versions contain prompts to elicit various kinds of descriptions from users. We also added features to streamline data entry, like suggesting tags based on geodata from external sources, commonly applied tags on MeLo, and suggested tags based on a user's previous tagging behavior. Several new functionalities that the users felt were imperative were also added, such as search and filter mechanisms based on date, area, and tags.

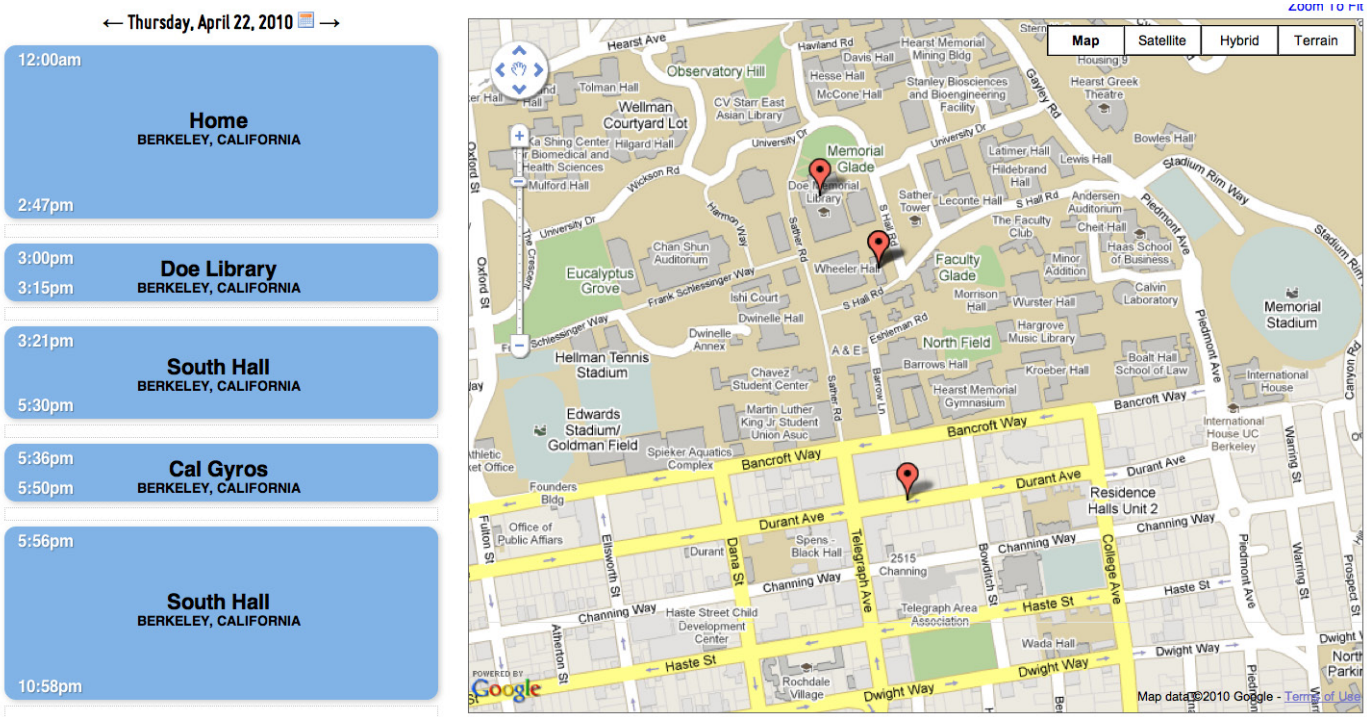



*Figure 3.* The main interface for describing location within MeLo shows a daily view that lists the user's location history alongside a map showing the places visited.

The main tagging interface shows a journal-like view, displaying the location history for one day. Users can browse through their historical data using the next and previous arrows or by picking a date in the calendar. All the places a user visited during the day are shown alongside a map that provides an overview of the day and gives the user a sense of his or her general whereabouts. The size of the blocks for each place is related to the length of time spent there.

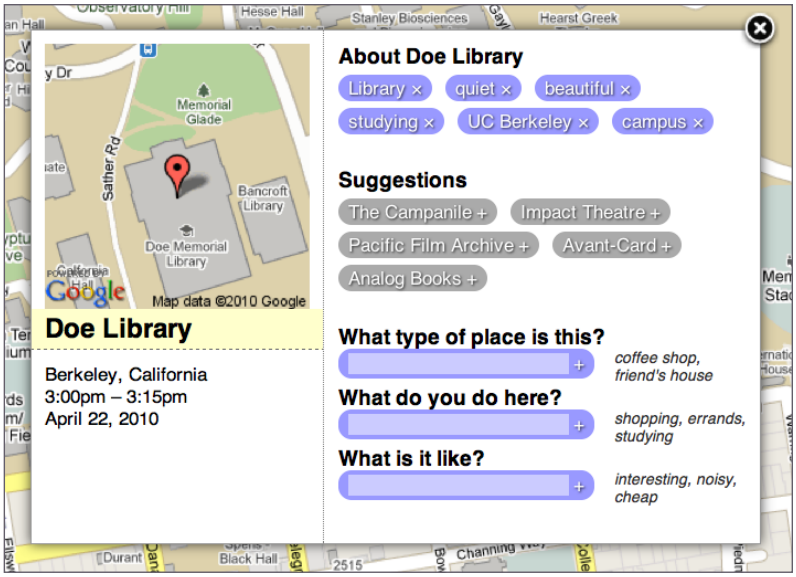



*Figure 4.* Location description interface.

When the user clicks on a specific location, a window appears with a zoomed-in view of the place and fields to describe it. The location name is shown, if known, otherwise it is labeled as an "unknown place" to encourage the user to provide a name. Location names can always be changed so the user can name a specific place according to what he or she calls it, as opposed to what it is called by everyone else. Geographic names for the location (city and state) are retrieved automatically and displayed.

On the right, a multi-part tagging interface is shown with the suggested tags. Clicking a suggested tag saves it for the location. Below this is a series of text fields which prompt the user with three questions: What type of place is this? What do you do here? What is it like? Entering text in these fields stores the descriptions with the location. The prompts are designed to encourage users to describe places in multiple dimensions instead of just entering a single name.

An earlier iteration of this interface had provided a single free-text box for tags. Although this was intended to allow greater flexibility and variety in tagging, we found that most users entered only the name of a place into this box. When users did enter categories, activities, or descriptions of place, our system was unable to distinguish them internally. Manually classifying all the tags in the first user test revealed four rough types: names (Sack's Coffeehouse), categories (coffee shop), activities (working) and descriptions (noisy), which coincided with findings from previous research.

In addition to the web interface, which is most useful for describing locations retrospectively, we developed a companion mobile interface for location tagging. This offers a simpler, two-step process of a name prompt followed by a single freetagging section.

The most significant finding during initial user testing was that showing the users a single textbox for a location, or even "Name" vs "Tag" fields in the case of the mobile application, did not effectively prompt them to enter more rich information about this location. Even adding example text in the interface did not encourage users to add more than the venue name to

the location. Users also found the tagging interface slow and awkward for entering data in quick succession, and not surprisingly, found the overall feature set to be lacking. The most common feature request was to add advanced multi-criteria search or filtering functionality, so they could more easily find a location previously visited, especially when they did not know enough relevant details about it to rediscover the location through a more traditional online search.

Our second round of interface brainstorming came up with everything from a speech bubble-style "conversation" to a "MadLibs," fill-in-the-blanks form. In the end, this interface was the most substantial change as a result of our user testing.

## Tagging Infrastructure and Integration with External Services

MeLo is focused on allowing people to create meaningful location histories. Adding a layer of more meaningful location, however, does not reduce the importance of basic geographical and venue names. Geographic data and venue names permit automatic annotations of locations imported into MeLo, and allows it to support the user in the process of tagging or categorizing their historical data. MeLo depends on existing services that provide traditional reverse-geocoding and location-based search to get geographical and venue information. Various location-based service providers have collected or acquired databases of venues, placenames and geographic hierarchies over time, and we leverage these services to provide such information. Using APIs provided by these services allowed us to focus on the core functionality delivered by MeLo while supporting venue search and traditional reverse-geocoding at a level on par with other services.

## Geographic Reverse-Geocoding and Hierarchies

When location points are imported in MeLo from services such as Google Latitude or Yahoo! Fire Eagle, there is little metadata attached to them. We leverage Yahoo!'s GeoPlanet API to retrieve geographic data such as city, region, state, and country. This data is automatically attached to each stay, allowing queries such as "all stays from Berkeley, CA." Additionally this data is essential in enabling the user to choose the granularity of the geographic location to share.

Additionally, each point imported into MeLo is assigned a Where on Earth Identifier (WOEID) retrieved using Yahoo!'s GeoPlanet API. WOEIDs are part of a canonical naming system created by Yahoo! to identify any location on Earth. WOEIDs are assigned to cities, towns, popular sites, and buildings, cities, regions, countries, continents, and other geographic features. WOEIDs are hierarchical and Yahoo! GeoPlanet allows queries for parents, children or neighbors of a given WOEID. They are gaining popularity among various services, most notably Twitter (Krikorian 2010).

We likewise integrated with a specialized reverse-geocoding service, GeoLenz, that is extending the scope of traditional geocoding. GeoLenz is a startup that is focused towards mapping areas such as university campuses, theme parks, stadiums, and

other large places that are not extensively covered by traditional geocoders. For example, in the case of South Hall, GeoLenz returns the following information: “University of California”, “Berkeley”, “Main Campus”, “Inside of building South Hall”, “building”, “School of Information.”

## Venue Information

To make it easier for users tagging stays in their location histories, MeLo fetches data about venues around the location of the stay from other sources. Currently we use the Yelp, GeoAPI, and Google Location Search to get data about venues around given locations. In order to effectively provide a low-latency experience to users, this data must be cached ahead of time on our servers. However, this may be in conflict with venue information provider terms of service, which may prohibit long-term storage of their data to prevent competition or whole-sale duplication of their proprietary data. This tension between technical requirements and legal concerns is not uncommon, but does provide a particular challenge in providing contextual data with sufficient speed for users.

## API

A public, RESTful API was designed to allow external developers to create applications that make use of information from MeLo. After we completed the first implementation of our project, we identified the resources in our system that clients would need to manipulate in order to have a robust collection of tools to work with. We then mapped operations for each of these resources to HTTP verbs (GET, POST, PUT, and DELETE), and specified required and optional parameters. The primitive resources we expose are points, stays, tags, and rules. We allow aggregate access to a particular user’s

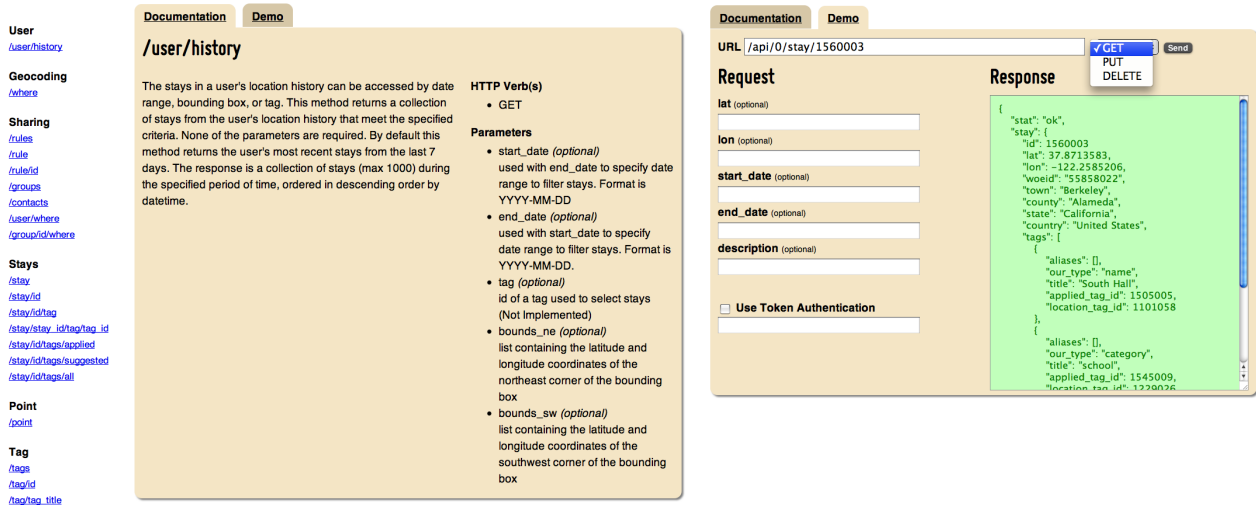



*Figure 5.* The MeLo API explorer combines documentation with an interface for testing calls to the API.

stays via the user's location history. For the purposes of sharing location with others, the system provides access to a user's contacts and groups of contacts. Finally, we provide a geocoding and reverse-geocoding system that works using a given user's location history and personal names for places.

Access to the API is authenticated using either a session key stored in a cookie, or via an access token and signature generated using a secret key. The token and secret may be obtained via an OAuth exchange. When the API encounters an error, it returns appropriate HTTP status codes. For example, if the calling user has not authenticated properly, the service responds with a 401 Unauthorized error; if the calling user does not have permission to access the requested resource, the service responds with 403 Forbidden; if the request is malformed or missing required parameters, the response is 400 Bad Request. Responses from the MeLo API are returned as the lightweight data-interchange format JavaScript Object Notation (JSON), making consumption simple. To facilitate data mashups in the browser, our API can also return responses as JSONP.

We also created an API explorer (see figure 5) to help developers learn about the API and debug calls. The explorer lets developers read documentation for the available methods alongside a framework for executing test API calls. This artifact was based on Flickr's API explorer, which provides similar functionality.

To test the robustness of our API, we re-implemented our own web interface as a client that uses public API methods, as we had originally built our interface by using privileged communication channels with the server. For the second version of our application, we wrote an API kit in JavaScript that provided access to the public methods exposed by the MeLo platform. By using our own API in the second implementation of our interface, we learned about certain methods that were missing yet required for the functionality we wanted.

The MeLo hackathon was valuable for obtaining feedback from developers, who had generally positive comments about the features, documentation, and flexibility of the API. Feedback from this event helped us to change the structure of response documents from the API for better consistency, and exposed additional bugs in functionality like improper return values and authentication issues. Though the brief period we provided for actually building projects didn't yield any finished products, the response to the API, documentation, and the API explorer was overwhelmingly positive.

## Privacy and Sharing

Because location data is often sensitive and personal, privacy has been a foundational principle of our project. Location data can be personally identifying (since, for example, very few people travel from my home to my school each day), reveal personal information about its subject (which bank I use, who I associate with and potentially even my sexual orientation—whether I frequent gay bars—or medical conditions—what health care specialists Planned Parenthood clinics I visit), and expose

the subject to physical intervention (a stalker could find me or a law enforcement agent could arrest me). It has been noted that extended location history information increases these potential risks (Maier 2010; Morris, Jr. 2010).

On the other hand, sharing location information provides real potential value to users. Usage of location-sharing services is growing rapidly, with Foursquare boasting of more than one check-in per second (Siegler 2010). Location can also provide some insight into context (what I'm likely to be doing and whether I'd like to be contacted) that users wish to selectively share with their friends.

Our meaningful location ontology is intended to allow for sharing that is both more useful and more protective of user privacy than existing services that tend to provide all-or-nothing revelations of latitude/longitude coordinates (e.g. Google Latitude) or venue name/address (e.g. Foursquare). Revealing to my family that I'm at a coffee shop rather than at the intersection of College and Derby Avenues in Berkeley gives them a better idea of where I am and what I'm doing without revealing precise information that could be used to find me.

Much research has been done into selectively sharing location information based on geographic location, recipient, or time of day (Sadeh 2010). Fire Eagle has a notion of hierarchical disclosure, which allows users to expose their exact position, neighborhood, city, state, and so forth. This happens on a per-application basis, rather than a per-user basis (e.g. Google can know my neighborhood, my blog can know my city, and Flickr can know my exact location). Studies exploring users' privacy concerns and behaviors in relation to storing location data have pointed out that people are often more comfortable with sharing abstract data with others (Consolvo 2005). Another research project has explored how people share location information in conversations for various reasons (Bentley 2008). We believe that MeLo's location sharing features are designed with a strong focus on how people already share information about where they are. MeLo users may reveal different levels of information (exact coordinates, name, category, activity, city/state) to different groups or contacts and under certain conditions (when in a certain type of place, or for a limited period of time).

### *Utilization of Context People Already Know*

People already have a lot of information about the routines and activities of others in their social network. Instead of assuming that all information is provided by MeLo, we have designed sharing mechanisms to leverage understandings that people have about their contacts. Providing simple abstracted data usually allows people to infer other rich information such as location and activity. However, since such sharing mechanisms require people to have prior knowledge, it safeguards against many kinds of privacy concerns relating to aggregation. It provides the possibility for plausible deniability (Akoi and Woodruff 2005) and does not reveal information to strangers.

Our sharing application allows for limiting the kind of information revealed about the users location to category (school, cafe, friend's house), city and state (Berkeley, CA or Seattle, WA) and will also extend to activity (reading, writing, working).

Stays in MeLo at the same location can be assigned different category, activity, and description tags at separate times. Allowing people to attach tags to their location allows them not only to share limited information, but also creates various cues to create rich social presence. Visualizations built on MeLo utilize this contextual information to provide meaningful displays of one's history. Sharing these visualizations with friends would provide interesting conclusions about one's past, without revealing the absolute location information used to create them.

### *Push vs. Pull*

We believe that users commonly recognize and value the distinction between making personal information available and broadcasting that personal information to their social network, even when the designers of information systems claim not to. In the case of the Facebook News Feed, for example, Facebook changed the distribution mechanism for changes to user profiles: a change to your profile would appear on a feed of each of your friends, where previously your friends would have had to check your profile each day to see any changes. When a large user protest broke out about the new feature, Facebook CEO Mark Zuckerberg expressed surprise and justified the feature by pointing out that no information that wasn't already publicly available had been made public. This response failed to recognize the differences users see in whether their friends really do read information about them, the different meanings that the same fact may have to different recipients, and the sociological norms for notifying contacts and, just as importantly, not notifying others of irrelevant information (Doty 2009; boyd 2006). The desire to avoid annoying contacts with irrelevant information was also explicitly cited in our user interviews as an important concern in discussing privacy rules.

In order to avoid that surprise, MeLo makes a clear distinction between what information about your location is available to different groups of friends and what information will be pushed out to them. We recognize the value of pushing changes in location information: for coordinating meetings, enabling favors, alerting friends to social events, etc., but note that it will often be a different level of information than what you'd be happy to tell a friend who asked. When creating rules for sharing location information in a certain situation and with a certain group, users can explicitly specify whether they want to let their friends see a particular information or want to tell their friends about it.

Because MeLo is providing a platform to enable other services, including location-sharing services, rather than functioning solely as a location-sharing service itself, the distinction between push and pull can't be controlled purely by our own

service. As a result, there is no purely technical restraint we can use to prevent a rogue client from accessing our API to request a user's location (pull) in an automated fashion, and notifying a particular recipient of a change. Instead, we require a parameter for each API request for a shared location, specifying whether that request was initiated by a user or not—if it was, we return the latest pull information, if it wasn't, we return the latest information that the subject has chosen to push to the recipient (in this way, a client can choose to re-implement our push functionality by polling). Though there are no technical means to prove that a client isn't lying with this parameter, clients that we find have lied about it can be removed for violating the terms of service.

This isn't the first or only example of specifying non-technical (and therefore not enforceable technically) policy in API parameters. The most recent version of Google's Map API requires developers to specify whether a sensor (like GPS) is being used to determine the user's location in all requests for their maps. The IETF's GEOPRIV standard requires that recipients of location information follow attached machine-readable rules for usage and retransmission of the subject's personal information (Tschofenig et al. 2006). The nascent Mozilla Privacy Icon Project (Raskin 2010) proposes a machine readable iconic representation of a privacy policy, much like the Platform for Privacy Preferences (P3P) before it (Cranor 2002). As in these examples, we believe that an API parameter for policy allows for a clear expression of policy and how it interacts with technology and enables enforcement through other means (namely, law).

### *Encrypting the Datastore*

To ensure privacy, ideally the user should be able to hide their data from the cloud, and location data should be encrypted while providing the desired features. However, encryption makes the data opaque to the server, and many of the features currently offered by the MeLo platform cannot be offered if the data is encrypted. We have explored various techniques and policies that would enable us to encrypt parts of the datastore while still providing services and acceptable response times. This section describes some methodologies described in literature, and our thoughts towards eventually offering the users the option to encrypt data stored with MeLo.

Encrypting users' location histories that are stored on the system provides security from unauthorized access and subpoena requests. There are, however, some challenges towards securing location data. Encryption adds additional burden on the users by expecting them to manage keys. Also, it constrains much of the background processing done on location data to provide services. Encrypting the data slows down response rates, especially on mobile devices that have limited network bandwidth and processing speeds.

Supporting features that require extensive pre-processing of location information such as adding metadata to points imported from other services, and generating complex visualizations would not be possible if all data is encrypted. Therefore we

propose methods of partially encrypting data, and storing multiple encrypted indexes that would allow specific kinds of queries while protecting user anonymity. We are still refining models that would allow us to encrypt our datastore while providing the current features.

Our proposed model for securing the user's data is to encrypt data where possible and to anonymize it elsewhere. Location information entered into the system would be stored unencrypted but not linked to any user identity, thus masking location among histories of other users (k-anonymity). This would include points, stays, and tags. For each user the database will maintain multiple encrypted references to points, stays, and tags using asymmetric cryptography. RSA public-private keys are generated for each user when they register. The public key is stored with their account while the private key is encrypted using a symmetric cryptographic algorithm (AES) before storage. Only the user can access the private key using their password.

The trust model for the server is assumed to be honest and trustworthy, but not a safe harbor. This means that though the users would trust the server to abide by the policies it publishes, the server can only be partially trusted with the location data of the users. The server abides by its policies, but those are not assumed to be same in the future. The server is not assumed to be completely secure, and it could be compromised. Also the server (in future policy changes) could inappropriately take advantage of the stored data. In such cases the users' location history should not be revealed and they should be able to revoke the server's access to future locations.

There are two primary modes for storing users' data:

- The current location of the user.
- All previous locations, or the user's location history.

The current location of the user is stored in multiple encrypted indexes that allow for easy range and location data queries. This is done to enable the location sharing features provided by the system. Location data is also added to a location store without any information relating to the user to provide for k-anonymity (Sweeney 2002). A pointer to the location history of a specific user is stored in an RSA encrypted store.

MeLo imports data from location services that do not encrypt user's data, so MeLo would encrypt this data as it is imported. Together with this encrpytion step the system could index data in a way that optimizes future queries.

*Data-driven queries:* For sharing using the pull mechanism, the server is provided with a list of users whose locations are to be retrieved. To serve this query, the last location would be encrypted with the respective public keys of recipients that are allowed access to a subject's location at the specified abstraction level (Khoshgozaran and Shahabi 2009).

*Space-driven queries:* For queries which request the identity of users in a specified location range (range or n-nearest neighbor queries) a separate index is required. This index stores the encrypted identities of subjects that have shared their

detailed location for each recipient using their public key in a sparse geohash. The encrypted results are sent to the user when a range query is made, which can be decrypted by the client. Then, the locations of each of the subjects can be retrieved using data driven queries (Khoshgozaran and Shahabi 2009).

*Space-driven queries for history:* Apart from creating indexes of contacts sharing data with a user, there is also an index of the user's personal location histories. The user's stays are encrypted and stored using sparse geohash for allowing range-based queries on their location history data. It would be essential to populate certain geohashes with dummy data that would be rejected when decrypted to preserve user anonymity.

*Time-driven queries:* Since we are encrypting the relationship between a user and their location history, it would be very expensive to return the user's entire location history to the client for description. Therefore, the relation between the user and their location data is stored for each day, month, and year information. This allows queries to be made that request data for a specific date, month, or year. The encrypted data is sent to the client for decryption.

## Third-Party Applications

### *meloscope*

meloscope is a web-based application that visualizes a user's location history, obtained through the MeLo API. It allows users to create categories for their history like "social," "health and fitness," "travel," "work," "leisure," and assign corresponding personal tags to these categories. These personal categories and tags are used to generate visualizations that provoke insights about a person's location history and general behavior. We interviewed users to learn how they currently review where they have been and what they have done during specific periods in the past. Based on users' responses, we determined that people normally

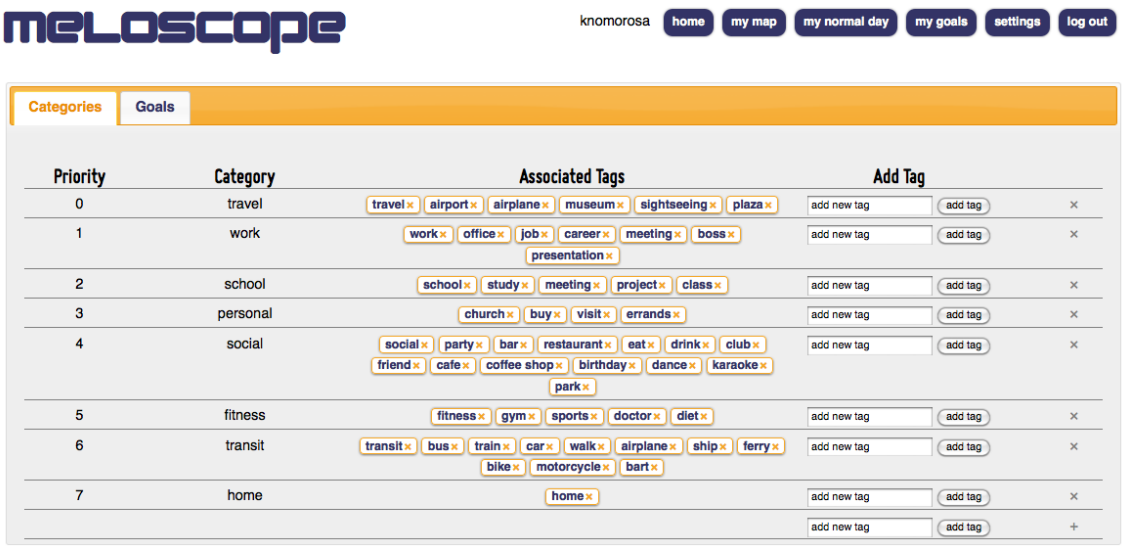



*Figure 6.* Meloscope allows users to sort their descriptions of locations into custom-defined categories.

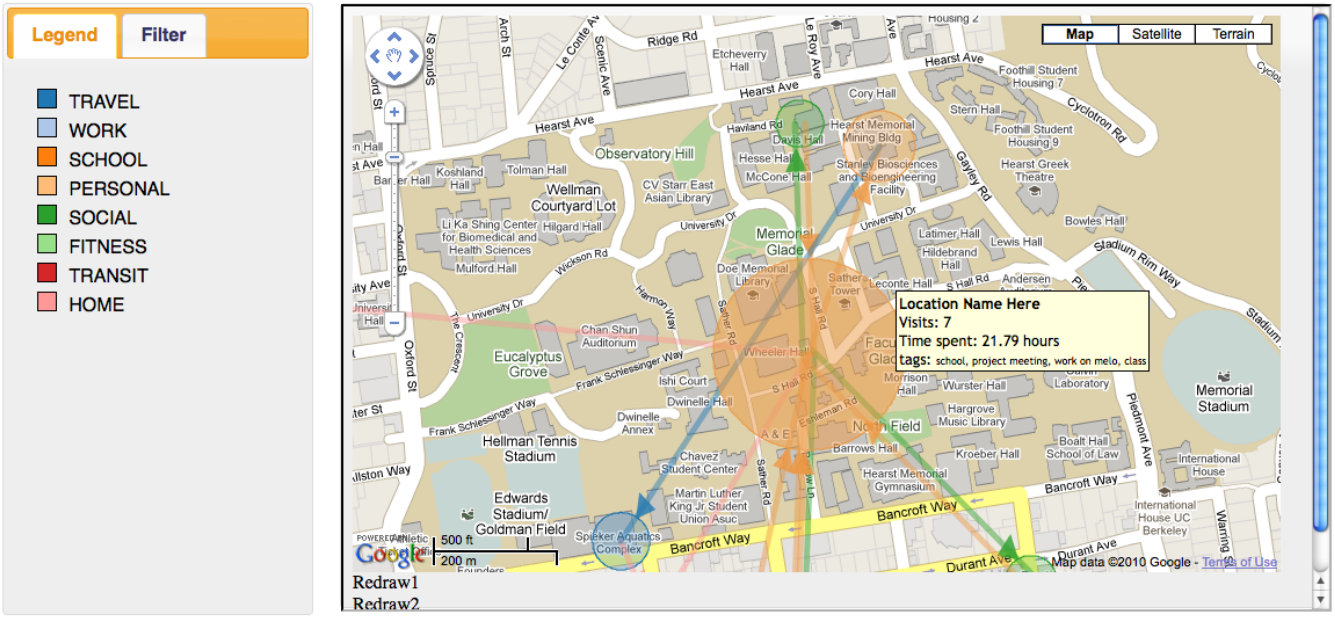



*Figure 7.* The map-based visualization in meloscope shows users the amount of time they spend at at locations and the frequency of trips between locations.

perform self-reflection to compare their personal history with friends; to remember what they have done, where they have been, and with whom; to remember special events or exceptions to the normal routine; and to review if they have met certain goals. We developed four different visualizations focused on remembering where users have been, special events and exceptions, and goal evaluation. These include a map-based visualization, a stacked line graph visualization, a calendar-based visualization, and a bar graph visualization.

The map-based visualization shows a user's location history overlaid on a map. Color-coded circles show visits to a particular location, with color encoding the location's classification, and size encoding the time spent at places. Paths between these locations show how often the user goes from the first location to the second location with the thickness of an arrow. Users can filter their data by path or based on a date range and specific days of the week to indicate which data points should be included. Filtering by path will only show locations that follow a certain pattern. For instance, the path "school-food-school" will show instances where the user first went to a place labeled school, followed by a place labeled food, followed by a place again labeled school. This allows users to determine their normal patterns and spot outliers in their behavior. Other interactions include hovering over map points, which reveals details about the location and filtering based on categories.

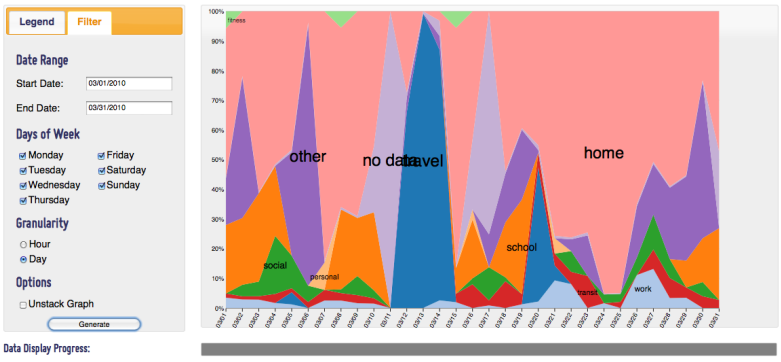



*Figure 8.* A stacked line chart shows how people spend their time throughout the day.

The stacked line graph visualization, on the other hand, aggregates the user's location history and

shows a summary of a user's activity in an hourly or daily granularity. This quickly shows what is "normal" to the user—for instance, that the user is normally home by 9:00pm, and shows some of the outliers, such as when the user is out socially at 1:00am on a certain day. Like the map visualization, users can filter based on date range and day of the week. They can also click on a specific category and see a break down of that category. For instance, clicking on 'social' may show that the user was at a 'bar,' 'party,' 'movie theater,' 'birthday party,' 'friend's house,' and the like. The user can also choose to unstack the line graph for easier comparison within the same time unit.

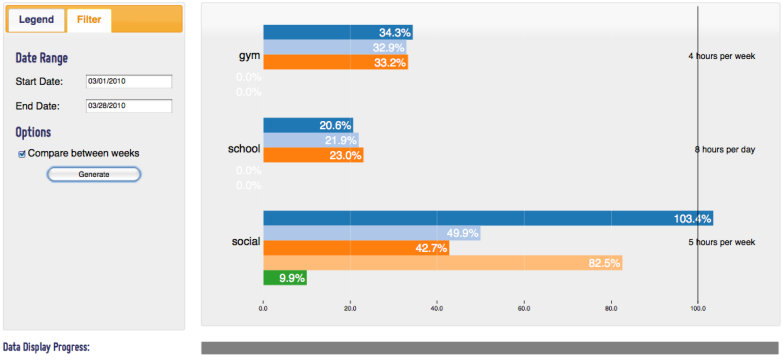



*Figure 9.* The goals visualization lets people track their progress against custom goals.

Users are also allowed to create and set-up goals, specifying which tag to look out for and what goals (in hours) are associated with that tag. A bar graph is then generated to compare if goals have been reached, and how much time a person needs to spend on an activity or location to meet that goal.

Finally, we developed a calendar-based visualization to identify patterns in users' location histories that occur on short-term scales (Greenberg 2010). This tool, called sparkcals after Edward Tufte's sparklines, take advantage of people's familiarity with the shape of calendars in a small multiples display. Each day is divided into user-defined bands which are shaded if a person

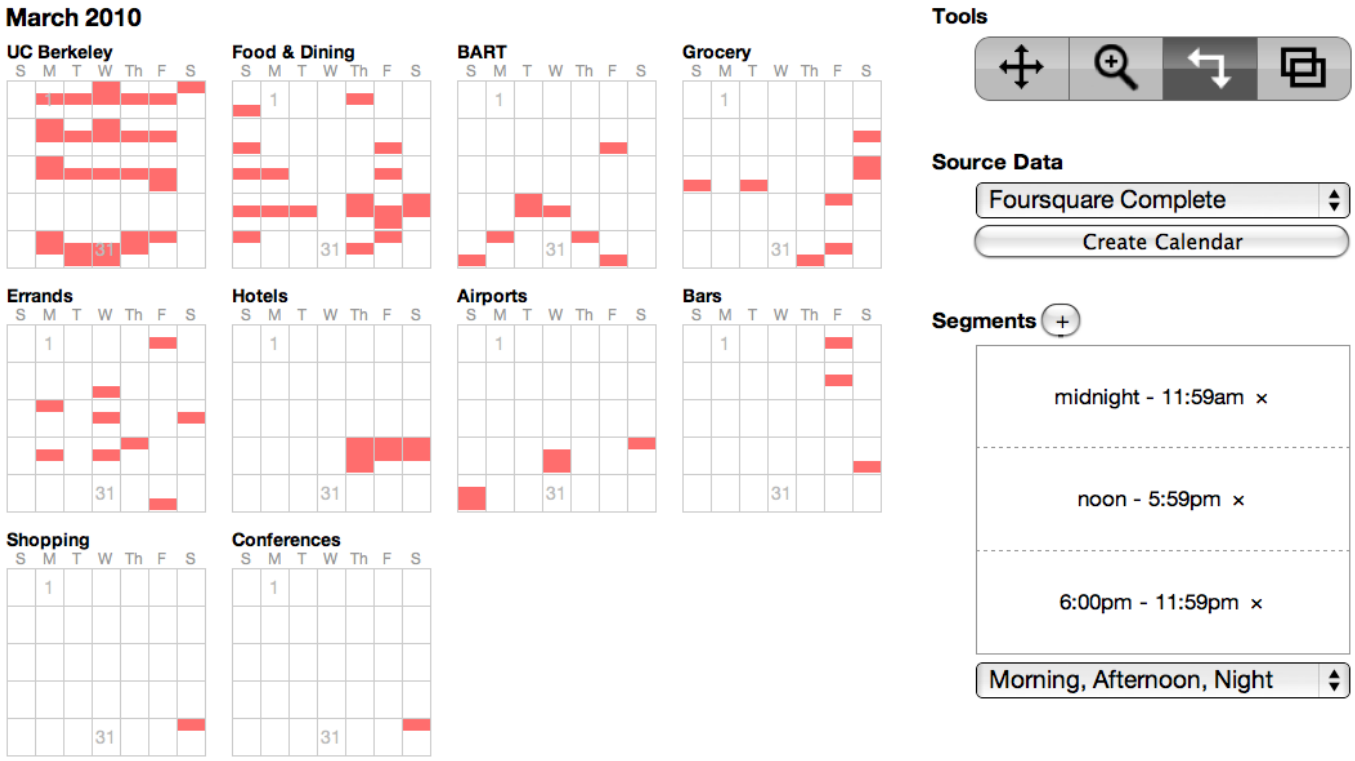



*Figure 10.* Sparkcals are a small multiples display of a single month that helps user compare data week by week.

was at a location within that time period. This results in a calendar which shows at a glance how and where a person spends most of his time in the specified time period. The absence of an otherwise dense set of bands indicates outliers clearly. Users can zoom into hierarchical sets of calendars, or pivot between dimensions to see groups of calendars. Users can also combine calendars to compare data in a single space.

meloscope leverages the personal tags inMeLo by going beyond visualizations based on purely geographic data. It brings a layer of abstraction enabled by the user's personal perception of place and creates visualizations that make more sense and meaning to the user.

### *ShareWhere*

ShareWhere is a mobile application that lets users create rules to communicate where they are to their contacts. This app provides an interface to the MeLo API that lets users set the conditions upon which other people can see their location and the level of granularity they receive. Users can express different levels of detail about their location to different people: friends from college can see the name of the city I'm in, close friends can see the name of the place I'm at, and my significant other can see my exact location. One focus of this project was to leverage the personal names people give to places both as what is seen by outside users and as the trigger for different levels of disclosure. In addition to letting friends see the names I have given places (e.g. "home" or "work" instead of my exact position), ShareWhere lets users use these names as triggers for different levels of

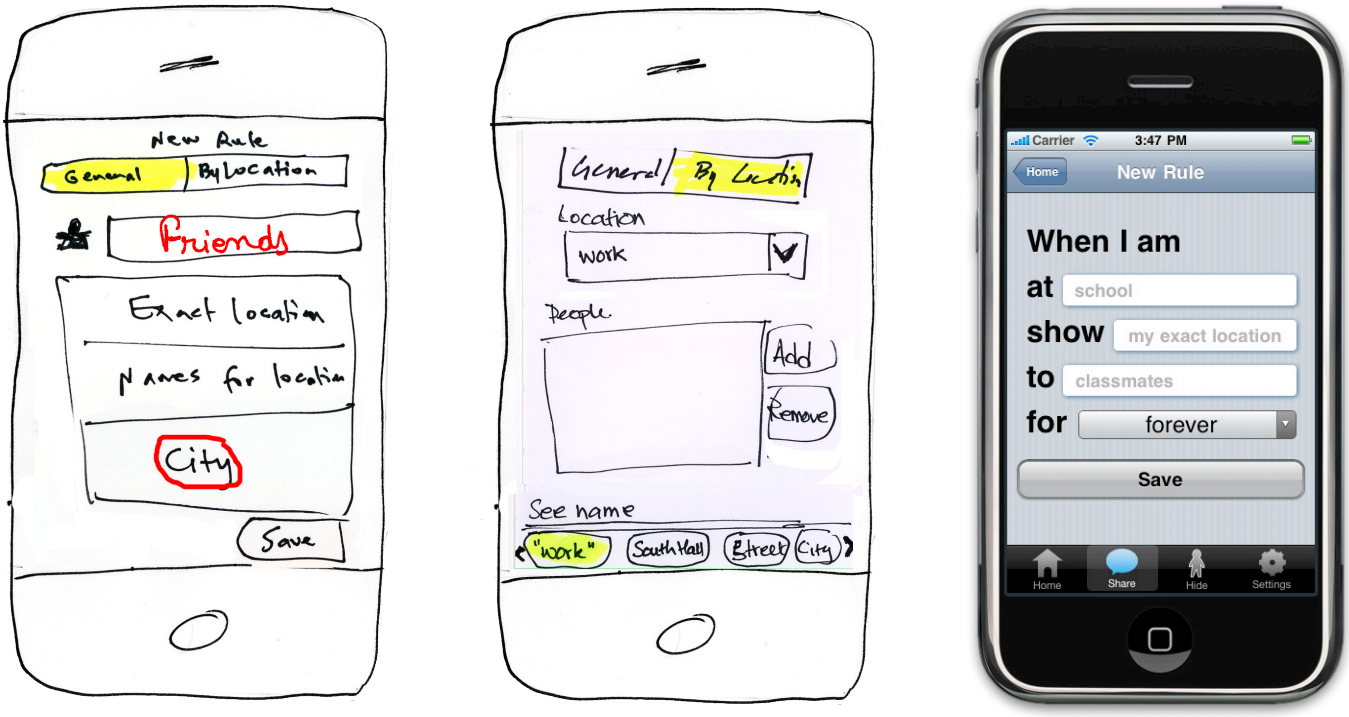



*Figure 11.* A single text box did not prompt users to enter rich information about a location, so this was changed to a multi-field interface in later revisions.

disclosure. For example, when I am at school (a name), let my classmates see my exact location.

Initial testing of the Sharewhere interface showed that users had difficulty with the rule-based interface for configuring sharing parameters. In testing we asked users to express what kind of location information they would want people in different social groups to see. We evaluated our interface based on how well it allowed users to codify these rules. User testing showed that an initial design where separate panels were used to create a rule was hard for users to understand, leading us to design a wizard-like approach where users fill in the blanks of a sentence to create a rule. In subsequent testing, users had much less difficulty creating sharing rules with this interface. Additional research in computer-supported cooperative work (Ackerman 2000) reveals that people deal with information disclosure on an exceptional basis rather than using pre-defined rules. As an attempt to overcome this issue, we included modes where a user could decide in a moment to immediately share information with another person or to hide his or her location.

Given the potential sensitivity of location information, this application was designed to minimize the likelihood of accidental disclosure by using smart defaults and making users aware of what information they are sharing. By default, sharing the user's exact location is limited to a short period of time, 30 minutes, so that the user has to take additional steps to make this information available on a long-term basis. In addition, ShareWhere lets users see what information other people see about them. This presents users with an opportunity to evaluate whether a sharing rule has the intended consequences and modify it.

ShareWhere also provides an easy-out to let the user disappear entirely from the system. A prototype version of the application supports prevarication about the user's whereabouts, allowing a person to indicate she is somewhere other than her current location.

User research for this project confirmed the perceived distinction between push and pull models of information disclosure and emphasized the interest in sharing less than perfectly precise information, even amongst close family.

To avoid having to re-create tools for contact management (a complex and tedious task for both our developers and our users), MeLo allows for importing groups and contacts from Google Contacts. ShareWhere users can select from these existing groups and contacts when choosing whom to share their location information with.

### *Whenyouat*

Whenyouat is an mobile application that enables people to set up location based triggers for others in their social network. The application, designed and developed to run on the iPhone OS, allows people to create messages for others that are tied to specific locations. Recipients of the task are alerted about the task only when they are at a place that matches the target location specified in the task. Whenyouat aims at supporting cases such asking a family member to get groceries or creating a

memory trigger at a specific location.

One sample usage scenario for whenyouat is as follows. Alice wants to get some things from the grocery store for the party at her house tomorrow night. She is busy and would probably not have time to go to the store herself, but does not want to spam all her friends with the request. Using whenyouat, she sets up a task detailing the things that she needs, and ties it to grocery stores. She sends out this task to her friends and roommates, with the end time set for the following evening. Next day, one of Alice's friends, Bob, visits the grocery store near his apartment and while checking whenyouat, he receives the alert from Alice. Bob accepts the alert and sends a message back to the task with the list of items he will be able to purchase. This message will be visible to others who have already accepted or will accept Alice's task.

Whenyouat builds on top of MeLo's reverse geocoding that returns category information about a given location such as grocery store, coffee shop, etc. This allows people to create tasks that are not just linked to absolute locations. The backend for Whenyouat is built on Google App Engine and handles storage for tasks and messages.

The design of the application does not require any kind of location sharing between users. In case an absolute coordinate is specified by the task creator, recipients are notified if they are within a short range of the location. This evaluation does not require the client application to share location information with the server, and is computed locally on the mobile device. However, in case of category based locations such as grocery store, the current location of the device is sent to MeLo for reverse-geocoding. This request does not need to be attached to a specific user. The MeLo service returns venue names and categories of places around the users' current locations. Locations of these venues are matched to tasks assigned to the user, and an alert is presented when appropriate.

### *Search and Filter*

The MeLo web interface initially allowed for only a single day view of location data. However, one of the most common requests was to provide an advanced search or filter screen where users could look up a location. Many users were very interested in MeLo as a way to rediscover locations they had been to before, but could not remember full details about. For example, "show me all places annotated as 'bars' that I visited in December in New York" is a common type of query. A basic search interface was added where the user can select any combination of a date range, bounding box, and tags, and receive a list of matching locations.

# Data Modeling

## Relationship Model

*Figure 12* illustrates the relationship of different objects used in MeLo. As can be seen in the diagram, almost all entities are related to a user. A location point refers to a particular lat/lon coordinate where a user is at a particular moment in time. Multiple points are then grouped together to form a stay, which specifies a duration of time that a person stays at a particular location.

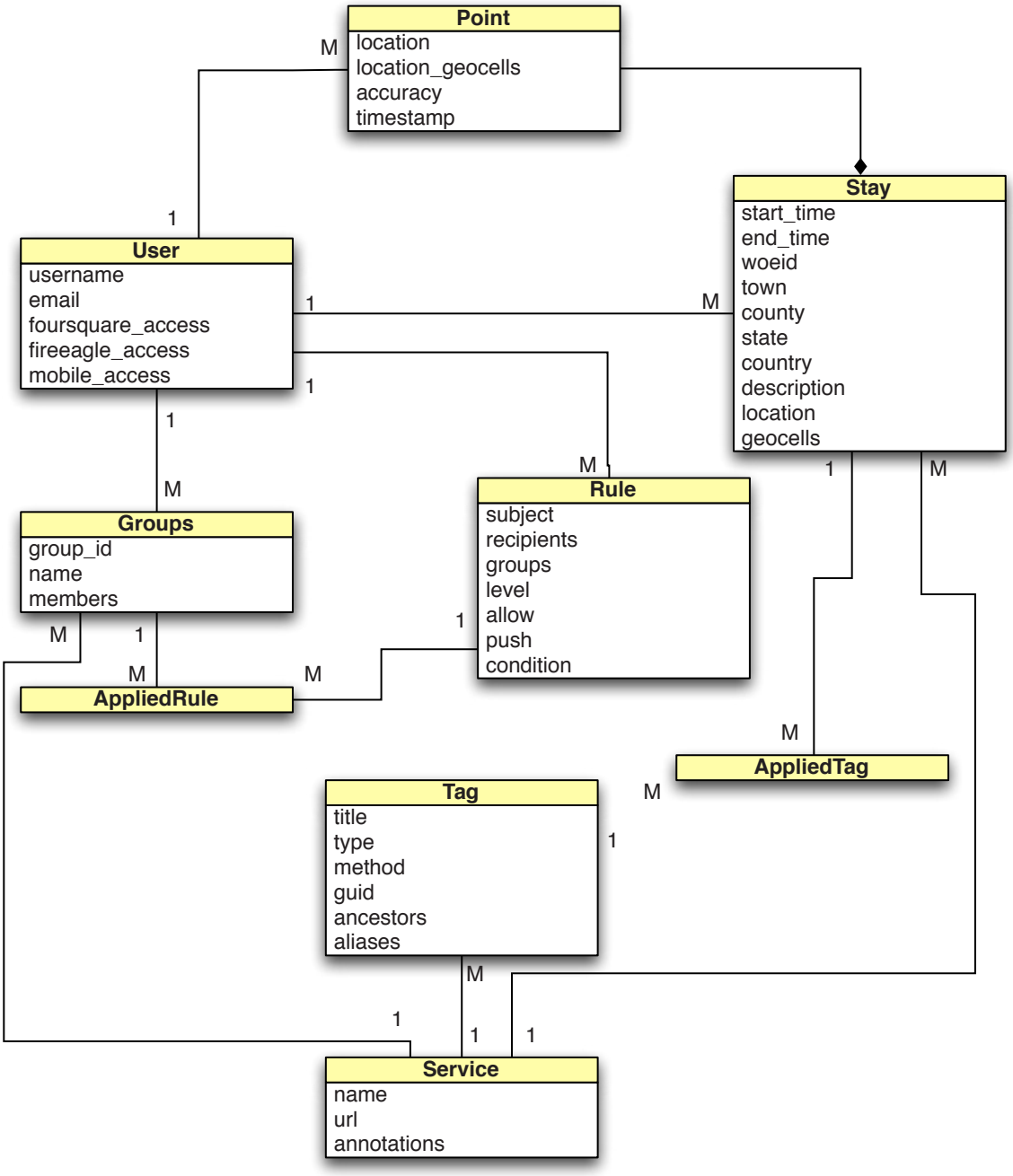



*Figure 12*. This diagram shows the relationship between objects in the MeLo data model.

A location tag can either be created by a user or applied based on suggestions from external sources, and these location tags are applied to a stay. A location tag can be applied to multiple stays, and a stay can have multiple location tags associated with it. The user sets privacy rules, as well as groups or contacts as recipients of the intended action (to share or to push location and to what level). As a new, more recent point comes in, it is evaluated against the privacy rules and the intended action is then executed. The service object keeps track of where tags, points, and stays are sourced from, whether they are user-generated, or whether they were taken from external services.

### *Modeling Stays*

During the first iteration of MeLo, points collected from Foursquare, Fire Eagle, and user-uploaded KML files were left as is. Geographic names for the locations were queried and applied to each and every coordinate gathered. Upon user testing of the original interface as well as during the development of more complex features such as tag suggestion and privacy, several issues with this model were discovered.

Services such as Fire Eagle can detect and send your information as often as users update them. GPS navigation devices and other background location services also provide a voluminous amount of data. This implies that if a person were at a specific location for an extended period of time, he or she could amass hundreds of data points for nearly the same coordinates, describing the same location.

On the user end, this means that they would have to apply their chosen tags repeatedly to all of those points for a single stay. This also results in the interface loading much more slowly than it should, as there is often too much information that passes between server and client. As a result, it takes too much time and effort to attach names to locations, defeating the purpose of the system.

Having too many points also resulted in the system having to process extensive, unnecessary information in order to execute processes such as suggesting tags and querying for geographic information. This led to extremely slow response times and the inaccurate analysis of user tags and user behavior.

One solution would be to group all of the points in the same geographic area as a location. Users could then add tags to these locations, which would apply to all the points at that location. We evaluated and discarded this option because it would not allow us to account for the changes in the same geographic space over time. Even for a single individual, places can be different depending on time. If you get a job at a department store, that place is now work in addition to a place you shop; a building on a university campus might be a classroom during the day and a theater at night; a girlfriend's house might become an ex-girlfriend's house.

A study by Hariharan and Toyama that involved the concepts of stays, paths, destinations, and trips provided a way to group points together (Hariharan and Toyama 2004). In particular, the concept of stays, a 'single instance of an object spending some time in one place', proved relevant as it would group together points in the same location for a specific duration into a single entity.

The concept of stays and the algorithm for grouping points into them were adopted and tweaked in order to fit MeLo's data collection model. As points are imported from the various services, MeLo takes a look at the timestamp for the point, queries the database to see if the point already belongs to a stay, either by it being within the start and end times of a stay within the same vicinity, or if it extends the duration of an existing stay. If MeLo determines that it does not belong to a stay, it creates a new stay with that point. In addition, if a point's timestamp is in the middle of an existing stay, but outside the set radius, the existing stay is split into two, and a new stay is created for the new point.

Unfortunately, the algorithm implemented does not yet determine the difference between stays and routes. It does not analyze if the person is in transit and is on his or her way to a stay. Instead, it represents specific points on a path as a very short stay. Given that the services used by MeLo to gather location information use a check-in model, and background location for mobile devices still have very low accuracy, we felt that implementing an algorithm for paths and other concepts in the Hariharan and Toyama paper, can be relegated to future implementations.

### *The Use of Geohashes*

Working with geographic data in a database presents performance issues because of its inherently two-dimensional nature. Simple queries like "which points in the database are within 5 miles of this point" are problematic. A naive approach to this problem would be calculating the distance between the target point and every other point in the database. The distance calculation itself (called the Haversine distance) is non-trivial. This approach does not scale well to handle thousands or millions of geographic entities, which is a requirement for a platform that would handle real-time data from thousands of users.

One solution to this problem is to perform dimensionality reduction to take the two-dimensional latitude and longitude coordinates and convert them to a one-dimensional space. This technique, called geohash, works by dividing the surface of the earth into numbered buckets. Buckets themselves are subdivided again and again into arbitrarily small cells. Each point is labeled with all the buckets that contain it. For example, South Hall, the building on Berkeley's campus that houses the School of Information, is located at 37° 52′ 16″ N, 122° 15′ 30″ W and has the geohash `8e63a2f5c08d7` in the implementation we used. This means that South Hall is in the `8` bucket, the `8e` bucket, the `8e6` bucket, and so on. Each successive digit of the geohash describes a smaller bucket, and we can use this to quickly find points in proximity to other points. If we want to find points within 100

meters of South Hall, we might search for the geocell `8e63a2f5c08d`; if we wanted points within 10 miles, we would use a shorter bucket name like `8e63a2f5c`. This technique allowed us to trade increased computation when adding data to the database for faster retrieval when searching.

The concept of geohashes is strongly used in the creation of stays, using a geohash to the ninth resolution (about 40 meters) to determine if a point is within the location boundaries of a stay. This is still augmented by a distance computation, however, for instances where locations exist in more than one bucket. In addition, geohashes are used in proximity fetches when searching for locations within a bounding box specified by the user, or when searching for nearby locations. This filtering mechanism is important when performing the geocoding and reverse-geocoding service that the MeLo API provides, as well as when suggesting personal tags.

This method of creating a tree and storing entities with a reference to all their ancestors in the tree is widely accepted as a best practice among location-based services. Exact details differ, but spatial databases from various providers (MySQL, Postgres, MS SQL, etc.) use a quadtree, R-tree, or similar data structure for geospatial indexing. A growing number of popular location-based services are using non-relational databases to store their geolocated data—Twitter and SimpleGeo with Cassandra, Foursquare with MongoDB—with the understanding that SQL joins would be prohibitively expensive. (WhereCamp discussion, April 4, 2010; Turner, J. 2010)

### *Effects of Varying Levels of Accuracy to the Data Model*

Since geolocation is commonly accomplished using a wide variety of techniques (GPS, A-GPS, WiFi triangulation, cell tower triangulation, IP geolocation, manual description, etc.) and under different conditions (indoors, outdoors, while moving, etc.), the accuracy level of a user's location history data may vary drastically. Services like Fire Eagle and Google Latitude largely abstract these distinctions from the user, accepting input from any geolocation method, and the Fire Eagle service doesn't track the documented precision of a particular data point.

As a result, the automated collection of location information about a user is likely to include at least some significantly incorrect data points. This can be frustrating to users trying to classify their location histories ("what was I doing over there?") and can easily throw off algorithms for automated sharing of information, with catastrophic results.

As technology improves, we can expect the accuracy of location information to improve as well. But in the short term (for example, background location tracking in the iPhone OS 4.0), information may continue to be very significantly inaccurate. The MeLo system actually has an advantage here: by using location history (where you usually go each day), we can better infer which points are likely to be spurious and what your precise location is in a crowded area. Though cell tower triangulation might

not be able to distinguish between the coffee shop on the corner and the hair salon next to it, if you go to the coffee shop every day and have never been to the hair salon, we can use that information to guess that you're probably at the coffee shop right now.

## Tagging Model

MeLo's tagging model is based on findings from previous studies conducted as well as observations made on test users' tagging behavior. Zhou et al. (2005) in a study on how people describe places conclude that there are five main classifications for location names: generic or functional (bookstore, grocery store), well-known public place (Starbucks, Target), specific public (Starbucks - Berkeley, Target - El Cerrito), personal / social (mom's house, home), and functionality / activity-based (shopping, class). Consolvo et al. in their study on location disclosure, on the other hand, showed differing levels of granularity that people use to describe where they are in certain situations (Consolvo 2005). Classification of generic places primarily deals with the functional aspect of place (residential, leisure, and service), with qualifiers such as specificity of function and privacy (Krämer 1995). Finally, a recent study shows how people classify locations based on a generic description (home, school), a functional description (gym, restaurant), and a business name (Starbucks, Target) apart from the more conventional geographic names (Lin et al 2009). The same study also indicates that people often create hybrid names for locations, bringing geographic, names, and personal descriptions together, such as "Starbucks in Berkeley".

In summary, these studies show that people commonly label locations in terms of where it is (Berkeley), that location's name (Starbucks), an activity associated with that location (studying), a function (gym, restaurant), and some combination of those. Observations on how our test users tagged their locations also reveal an additional dimension on how people often relate a certain description to places (e.g. romantic). Taking these into consideration, MeLo models tags according to *name,* which could both be community-accepted or business names for a location, as well as names that are personal to the user; *category,* which correspond to the functional description of the place; *activity,* which describes what the person is doing at the particular location at that time; and *description,* which are adjectives that qualify the place.

These annotations are applied to a stay as opposed to a set point or group of points, owing to the realization that time is another dimension to consider when talking about location and place. Some tags may be applicable to a location at one point in time, but not necessarily at another. For instance, UC Berkeley may be tagged as 'school' now, but may be tagged as 'work' sometime in the future when it applies. We believe that introducing this method for applying tags enables a host of possible uses, such as for self-reflection and location sharing.

Tags in MeLo may come either from an external service or be entirely user-generated. Tags are suggested to the user for a particular stay based on other tags that were applied near to that stay or on information from external services (commonly

the name or category of commercial venues from services like Yelp or Foursquare). Each tag may be a name, category, activity or description (see above).

MeLo seeks to consolidate tags where possible in order to provide more interesting and useful aggregations as well as calculations based on tags. To that end, some pre-canned categories (based on the classification systems of Yelp, Foursquare, and Gowalla) are provided along with a system of aliases so that, for example, "grocer," "grocery," and "grocery store" are all considered the samevz.

We also seek to provide a shallow hierarchy of category tags so that the system can infer, for example, that all Mexican restaurants and Indian restaurants are restaurants. These hierarchical inferences can be useful for various visualizations (grouping like things together) and sharing rules (sharing your location when you're at any type of restaurant).

### The Data Store

Optimization is particularly and inherently difficult for our applications because of the multi-dimensionality of our data. A geohash provides a linearization of two-dimensional geographic data, but much of our data also depends on the additional dimensions of time and user. Because places can have very different functions (imply different activities, even have different names) at different times of the day, week, or year (Blackrock City, etc.) and have different meanings to different people (this is my school but your workplace), MeLo can't easily take advantage of monolithic indices or large caches of frequently accessed data.

A continuing challenge for us is determining which data (geographic or venue-oriented, in particular) is common to all users and times and can therefore be easily cached and indexed, and which data needs to be calculated or combined per user. In order to take advantage of our non-relational database, we've had to diligently pre-determine the types of retrieval queries users want to make and create custom composite indices. We continue to analyze patterns of repeated database calls where batching queries or de-normalizing our data model can improve performance.

## Conclusion And Recommendations For Future Research And Development

Inspired by the inadequacies we found in existing systems for location tracking and sharing, we sought to build a location-based platform that would take advantage of the deep theoretical concepts of place and our everyday notions of where we are. No single software artifact can fulfill so lofty a goal, but we believe we have shown one step (personalized annotation of location history) towards that goal and multiple useful applications (granular location-sharing, visualizations of location history,

and contextual location triggers) that even that first step affords. Based both on those successes and the substantial challenges we faced in building the MeLo platform, we believe large opportunities exist for expanding this service, improving upon it, and extending the same techniques to other domains.

Although the current system relies solely on personal location histories as a source of location data, the same concepts of personal names and categories could equally apply to places a user hasn't been. For example, the Local Ground project (Tsai and Van Wart 2010) lets students describe their ideas about places in their town or neighborhood, including "bad parts of town" which wouldn't be a part of their own location history. Conversely, I might describe future vacation spots as places I want to go, even though I've never been there myself. Test users provided feedback on many additional useful features a future version of MeLo might provide: correcting points in their histories, integrating data from more and different services, and different ways of grouping points and stays.

Towards the end of a truly useful parameterized and personalized ontology of location, future work should look at more nuanced ways for determining when annotations should be shared, aggregated, or transferred. MeLo's categorization system could provide more intelligent conclusions by understanding indexical relationships to places ("my home" is "a friend's house" to a friend) or understanding communal ideas of place (this is "our school", which is neither universal nor uniquely personal). Our testers' experience with categorizing their own location histories has shown that it is possible to make fruitful use of personal names for places, but just as importantly that users may need prompting in order to categorize their own data in more than the most basic of ways. Statistical inferences from patterns of location history may enable more efficient automatic categorization of places and corresponding activities. History data may actually provide an effective solution to the accuracy problem of current geolocation technology—a major challenge for the effectiveness of our project and any location-based service—by providing a key heuristic for disambiguation.

Although the MeLo project has focused on building a system for tracking, categorizing, and sharing location history, the platform provides, in essence, a parameterized ontology of location. A place has different meanings for different people and at different times, adding at least one or two dimensions to the traditional model of simple geocoding. Our experience building MeLo has shown some inherent challenges in handling that multi-dimensional data: both in the technical details of designing a database to index and analyze that data in a different way for every single user, and in the usability challenges of prompting users to navigate and categorize the different facets of their location histories. We expect that adding meaning to location coordinates will not remain a purely academic result and the movements of large corporations in the location ecosystem suggest that they see this need. The wider adoption of location-based services outside of the technical community may in fact rely on a system for personal, meaningful descriptions of place.

We hope that with continued use MeLo will help more and more users track and classify their own histories using a vocabulary that is meaningful to them. And a real metric for our success will be the presence and value of applications built on top of our platform that take advantage of this personal context and nuance. More broadly, we see a rich ground for research in both social science and information science as users of MeLo and similar systems see the real-world implications of their own classifications of the places around them.

## Acknowledgements

We would like to thank our advisor, Deirdre Mulligan, for her insight and guidance on this project. We would also like to thank Erik Wilde for his comments and suggestions on MeLo's API design. Third-party applications that utilize the MeLo API would not have been possible without the contributions and hard work of the following people: Ephrat Bitton, Jonathan Chu, Matt Gedigian, Prateek Kakirwar, Niranjan Krishnamurthi, Aditi Muralidharan, Thomas Schluchter, Kate Smith, and Nathan Yan.

# Works Cited


Aoki, P. M, and A. Woodruff. 2005. "Making space for stories: ambiguity in the design of personal communication systems." Pp. 181–190 in *Proceedings of the SIGCHI conference on Human factors in computing systems*.

Ark, W. S, and T. Selker. 1999. "A look at human interaction with pervasive computers." *IBM systems journal* 38:504–507.

Barkhuus, L., and A. Dey. 2003. "Location-based services for mobile telephony: a study of users' privacy concerns." Pp. 709–712 in *Proc. Interact*, vol. 2003.

Bentley, F. R, and C. J Metcalf. 2008. "Location and activity sharing in everyday mobile communication." Pp. 2453–2462 in *CHI'08 extended abstracts on Human factors in computing systems*.

boyd, danah. 2006. "Facebook's' Privacy Trainwreck': Exposure, Invasion, and Drama." *Apophenia Blog* 8.

Consolvo, S. et al. 2005. "Location disclosure to social relations: why, when, & what people want to share." P. 90 in *Proceedings of the SIGCHI conference on Human factors in computing systems*.

Counts, S., and M. Smith. 2007. "Where were we: communities for sharing space-time trails." P. 10 in *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*.

Cranor, L., M. Langheinrich, M. Marchiori, M. Presler-Marshall, and J. Reagle. 2002. "The platform for privacy preferences 1.0 (P3P1. 0) specification." *W3C recommendation* 16.

Daniels, S. 1992. "Place and the geographical imagination." Geography-London 77:310–310.

Dey, A. K. 2001. "Understanding and using context." *Personal and ubiquitous computing* 5:4–7.

Doty, N. 2009. "Changing the audience of "existing" information." http://npdoty.name/papers/changing-the-audience.pdf.

Dourish, P. 2004a. "What we talk about when we talk about context." *Personal and ubiquitous computing* 8:19–30.

Dourish, P. 2004b. *Where the action is: the foundations of embodied interaction*. The MIT Press.

Eagle, N., and A. S Pentland. 2009. "Eigenbehaviors: Identifying structure in routine." *Behavioral Ecology and Sociobiology* 63:1057–1066.

Edwardes, A. J. 2009. "Geographical perspectives on location for location based services." Pp. 1-4 in *Proceedings of the 2nd International Workshop on Location and the Web*. Boston, Massachusetts: ACM http://portal.acm.org/citation.cfm?id=1507136.1507141 (Accessed May 7, 2010).

Gale, G. n.d. "Location is a Key Context, But Most People Don't Know This." http://www.vicchi.org/2010/02/12/location-is-a-key-context-but-most-people-dont-know-this/ (Accessed May 6, 2010).

Greenberg, R. "Sparkcals: Small Multiple Calendars for Detecting Temporal Patterns." http://ryangreenberg.com/papers/sparkcals.pdf.

Hariharan, R., and K. Toyama. 2004. "Project Lachesis: parsing and modeling location histories." *Geographic Information Science* 106–124.

Khoshgozaran, A., and C. Shahabi. n.d. "Private Buddy Search: Enabling Private Spatial Queries in Social Networks (PDF).".

Kirkpatrick, M. 2010. "Location Data and Privacy Subject of Congressional Hearing Next Week: Today's Top Stories on Geolocation." http://www.readwriteweb.com/archives/congress_to_hold_hearing_on_location_data_and_priv.php (Accessed May 6, 2010).

Krämer, B. 1995. "Classification of generic places: Explorations with implications for evaluation." *Journal of Environmental Psychology* 15:3–22.

Krikorian, R. 2010. "WOEIDs in Twitter's Trends." (Accessed May 6, 2010).

Lin, J., J. Hong, N. Sadeh, and Carnegie-Mellon School Of Computer Science. 2009. "Understanding People's Place Naming Preferences in Location Sharing.".

Maier, Fran. 2010. "7 Takeaways from the FTC roundtable in Berkeley, CA." http://www.truste.com/blog/?p=477 (Accessed May 6, 2010).

Malik, Om. 2010. "Will 2010 Finally Be the Year of Location?." http://gigaom.com/2010/01/10/2010-year-of-location/ (Accessed May 7, 2010).

Massey, D. B. 1994. *Space, place, and gender*. Univ of Minnesota Pr.

Merleau-Ponty, M. 1996. *Phenomenology of perception*. Motilal Banarsidass Publishers Pvt. Ltd.

Morris, Jr., John. 2010. *The Privacy Implications of Commercial Location-Based Services*. http://energycommerce.house.gov/Press_111/20100224/Morris.Testimony.2010.02.24.pdf.

Nielsen, J. 2000. "Why you only need to test with 5 users." *Jakob Nielsen's Alertbox* 19.

Quine, W. V. 1951. "Main trends in recent philosophy: two dogmas of empiricism." *The Philosophical Review* 60:20–43.

Raskin, Aza. n.d. "Making Privacy Policies not Suck." (Accessed May 6, 2010).

Sadeh, Norman. n.d. "Locaccino." *Locaccino*. http://locaccino.org/ (Accessed May 7, 2010).

Siegler, MG. 2010. "Foursquare Now Seeing A Check-In Each Second." http://techcrunch.com/2010/01/12/foursquare-check-ins/.

Smith, B. 2003. "Ontology." *Blackwell guide to the philosophy of computing and information* 166.

Snekkenes, E. 2001. "Concepts for personal location privacy policies." Pp. 48–57 in *Proceedings of the 3rd ACM conference on Electronic Commerce*.

Svanaes, D. 2001. "Context-aware technology: a phenomenological perspective." *Human-Computer Interaction* 16:379–400.

Sweeney, L. 2002. "k-anonymity: A model for protecting privacy." *International Journal of Uncertainty Fuzziness and Knowledge Based Systems* 10:557–570.

Tsai, J. Y, P. G Kelley, L. F Cranor, and N. Sadeh. 2009. "Location-sharing technologies: Privacy risks and controls." in *Research Conference on Communication, Information and Internet Policy (TPRC)*.

Tsai, Joyce, and Sarah Van Wart. n.d. "Local Ground." http://localground.org/.

Tschofenig, H., H. Schulzrinne, A. Newton, J. Peterson, and A. Mankin. 2006. "The IETF Geopriv and presence architecture focusing on location privacy." in *Position paper at W3C Workshop on Languages for Privacy Policy Negotiation and Semantics-Driven Enforcement, Ispra, Italy*.

Tuan, Y. F. 2001. *Space and place: The perspective of experience*. Univ Of Minnesota Press.

Turner, A. 2006. *Introduction to neogeography*. O'Reilly Media, Inc.

Turner, James. 2010. "Joe Stump on data, APIs, and why location is up for grabs." http://radar.oreilly.com/2010/03/joe-stump-talks-location-and-n.html (Accessed May 6, 2010).

Weiser, M. 1991. "The computer for the twenty-first century." *Scientific American* 265:94–104.

Zhou, C., P. Ludford, D. Frankowski, and L. Terveen. 2005. "Talking about place: An experiment in how people describe places." *Proc. Pervasive, Short Paper*.