

# Piggyback Prototyping: Using Existing, Large-Scale Social Computing Systems To Prototype New Ones

Catherine Grevet and Eric Gilbert  
School of Interactive Computing  
Georgia Institute of Technology  
cgrevet@gatech.edu, gilbert@cc.gatech.edu

## ABSTRACT

We propose a technique we call *piggyback prototyping*, a prototyping mechanism for designing new social computing systems on top of existing ones. Traditional HCI prototyping techniques do not translate well to large social computing systems. To address this gap, we describe a 6-stage process for prototyping new social computing systems using existing online systems, such as Twitter or Facebook. This allows researchers to focus on *what people do on their system* rather than *how to attract people to it*. We illustrate this technique with an instantiation on Twitter to pair people who are different from each other in airports. Even though there were many missed meetings, 53% of survey respondents would be interested in being matched again, and eight people even met in person. Through piggyback prototyping, we gained insight into the future design of this system. We conclude the paper with considerations for privacy, consent, volume of users, and evaluation metrics.

## Author Keywords

Social computing; large-scale systems; prototyping; evaluation; privacy; ethics; social interactions.

## ACM Classification Keywords

H.5.3: Group and Organization Interfaces. Evaluation.

## INTRODUCTION

We were contemplating building an app to introduce people waiting in airports (as illustrated in Figure 1). We quickly realized that attracting users would be a central challenge since the usefulness of the app rests on the fact that many others are also using it. How could we pair up people in airports unless many tens of thousands of users, at the very least, had opted-in? The uncertainty involved with scaffolding a social app to this scale could have stalled our project. Even massively funded and pioneering commercial systems (such as Google Wave<sup>1</sup> and Color<sup>2</sup>) have failed to achieve this critical mass of users. Thus, building the app was risky:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [Permissions@acm.org](mailto:Permissions@acm.org).

CHI 2015, April 18 - 23, 2015, Seoul, Republic of Korea.

Copyright is held by the owner/author(s). Publication rights licensed to ACM.

ACM 978-1-4503-3145-6/15/04 \$15.00

<http://dx.doi.org/10.1145/2702123.2702395>



Figure 1. Storyboard of the airport meet-up app we wanted to build to pair people while they wait.

it would cost significant engineering resources and ... what if no one used it? Did we have to build the system end-to-end just to see if we had a viable concept?

Of course, this is what *prototyping* is designed to solve. Prototyping can pinpoint fundamental flaws in interactive systems *before* a design team invests considerable energy building the system. Most HCI systems start as prototypes: they are designed and developed iteratively, and at increasing levels of fidelity. Unfortunately, existing HCI prototyping techniques do not translate well to social computing systems. For example, imagine if we had tried to “wizard-of-oz” [1] the airport meet-up app. We would have needed an entire team of “wizards” living the life of Tom Hanks in the movie “The Terminal.” They would have each had to buy plane tickets (so they could get past security), and then set up camp for days in airports across the country. Then they would have met a participant when they received a match, presuming they had not been arrested first.

Our solution to these shortcomings was to develop a novel prototype technique: *piggyback prototyping*, a 6-stage prototyping mechanism for testing and iterating on new social computing designs. It works by coupling semi-autonomous bots to already successful large-scale social computing systems, such as Twitter. Piggyback prototyping overcomes the challenges of obtaining critical mass by leveraging existing social platforms. A piggyback prototype can focus on

exploring *social interactions*, rather than *interface interactions*. Specifically, social interactions are those involving people directly interacting with each other. Such interactions can involve messaging or commenting, liking or up-voting, meeting face-to-face, sharing information to each other, etc. We are not using “the crowd” to add information or change the display of an interface, rather Piggyback Prototyping is used for systems in which people directly communicate with one another.

In this paper, we describe the mechanism and an example of deploying a piggyback prototype. We created a semi-autonomous bot that paired Twitter users who had checked-in to the same airport and told them to meet. We ended up forming 3,161 pairs, from which we received 576 tweet replies, 183 survey responses, and 8 participants who actually met in person. We learned that people would, in fact, meet others through our envisioned system, and more importantly, specifically how future design iterations could facilitate more meet-ups to occur.

This paper’s contributions are threefold. First, we describe the steps of piggyback prototyping, illustrating along the way with examples. Second, we describe our experience using a piggyback prototype in a real-life project. Finally, we reflect on the scope, benefits, and limitations of the piggyback prototyping technique.

## BACKGROUND

First we briefly review the history of prototyping in HCI. Then we describe the challenges faced by social computing systems and why traditional HCI techniques do not work for large-scale social systems.

### A brief overview of prototyping

In HCI, a prototype is “a concrete representation of part or all of an interactive system” [1]. This is in contrast to an abstract representation, such as a verbal description. As such, prototypes can be manipulated. Their manipulation creates a vehicle for communication between designers, engineers, and users [1, 13]. The prototype should serve a well-defined function, such as prototyping the role of a new capability, the look and feel of an existing concept, or the implementation of how a system actually works [13].

Through the process of adapting and refining the prototype, it evolves. To allow for this flexibility, prototypes tend to be rough and sketch-like. Low-fidelity prototypes are a rapid prototyping technique: they are easy and quick [1]. Common methods for this are *paper prototyping* where an interface is literally drawn on paper [22], or *wizard-of-oz prototyping* where the interface is a curtain behind which a researcher responds to user input [1]. *Higher fidelity prototypes* require significant coding, but provide a more finished look [1]. Crafting a relevant prototype requires the skill of a domain expert.

Prototypes serve to elicit guidelines for future design improvements and they are evaluated in observational settings.

To use Computer Science terminology, they are compared against “benchmarks of performance.” These metrics are determined prior to a study and guide the tasks that users will be asked to perform [1]. Other forms of evaluation are less directed: the user creates meaning through interacting with the prototype such as with *probes* [7]. In addition, prototypes can also be viewed as essential tools for generating design ideas and insights [16].

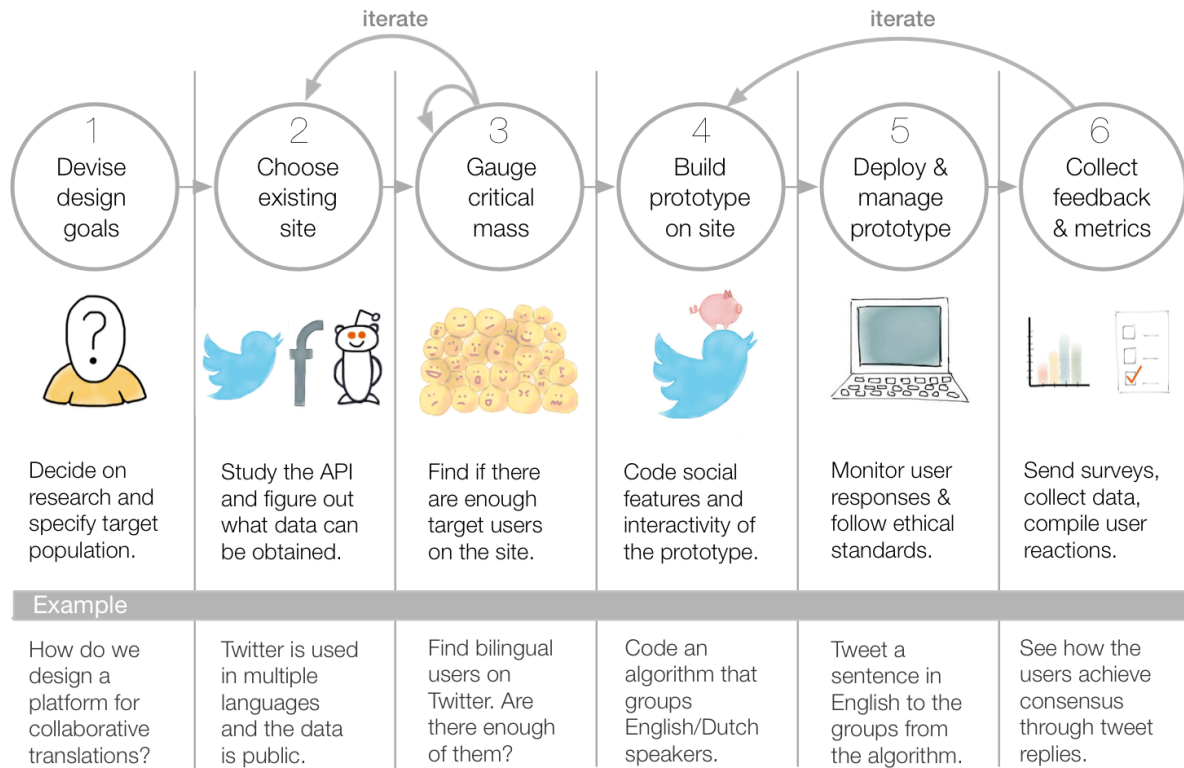
### Prototyping social computing systems

Here we use “social computing systems” as a broad term for technologies that involve human-human interaction. We distinguish between two types of social computing systems: those for small groups, and those for larger crowds. Small-scale collaborative group systems can be prototyped with adaptations of traditional HCI techniques. Social computing research has many examples of these [9]. For example, “paratypes” are probes that can help understand the social context and social acceptance for a new technology [14]. When using a paratype, a researcher surveys reactions to the prototype as they go about their day-to-day activities. To our knowledge, however, there are no prototyping techniques for large-scale systems.

A simple search in the ACM digital library underlines this gap: only 4 papers from the CSCW conference contain the words “prototype” or “prototyping” in their title, as compared to 85 papers for CHI. If we naively align HCI work to CHI and social computing work to CSCW, it becomes apparent that prototyping techniques for social computing are scarce. This may be due to a number of challenges.

One major problem with large-scale social computing systems is the need to obtain critical mass [10]. To avoid this issue, some systems have foregone testing with people and opted instead with agent simulations [6] or simply proposed a prototype without evaluation [18]. In these approaches, it is not possible to evaluate the social affordances of the system. Others have created complete systems in the hope of attracting users to a polished product. Turkopticon, a tool for Amazon Mechanical Turk (AMT) workers, for example used bootstrapping by collaborating with an AMT employer to obtain their initial users [15]. At the time of publication, four years after the launch they had 7,000 installations [15]. Some may be lucky, such as Link Different, a social trans-luence tool for Twitter, which gained 144,232 users in two months thanks in part to media attention [8]. In both cases, obtaining critical mass was central to their efforts. This may not be the case for all projects.

Another challenge is determining evaluation metrics. Is a system that gets fewer than thousands of users a failure? Number of users is a common metric for commercial products, and we have also—we argue to our detriment—adopted it as a standard for research. Would Turkopticon have been irrelevant if it only had 10 users? If achieving critical mass is our main measure of success, we are limiting the quantity and quality of social questions we can ask.



**Figure 2. The 6 stages of piggyback prototyping: 1) Devise design goals, 2) Chose existing site, 3) Gauge critical mass, 4) Build prototype on site, 5) Deploy and manage prototype, 6) Collect feedback and metrics. Stages 3 and 6 are iterative, if critical mass is not present for the target population (3) or the deployment was not successful (6) previous stages may be modified and re-evaluated.**

## INTRODUCING PIGGYBACK PROTOTYPING

We next introduce the six stages of piggyback prototyping, a social computing system prototyping technique that utilizes existing social platforms to evaluate novel social interactions for large-scale systems. Piggyback prototyping is best suited to projects that require coordination between multiple people and that do not have access to readily available large-scale social data. We believe, for example, that a large variety of social matching systems could have been prototyped with this technique, such as organizing people for disaster relief, assisting a collaborative activity, matching people according to interests, and many others. Piggyback prototypes involve 6 stages, two of which distinguish it markedly from other techniques: a non-social pilot (stage 3 in Figure 2) and a social deployment (stage 4 in Figure 2). The evaluation of the prototype will likely involve mixed methods in which a researcher might craft a survey, plan an interview, collect log data, or compile user responses.

### 1. Devise design goals

The first step is to decide on the research goals and the desired social interactions. Figure 2 presents an example for a collaborative translation system, a system resembling the existing site Duolingo<sup>3</sup>. We also suggest in this step to determine the target population for the prototype. This will be the basis for the next step and for planning the architecture of the prototype.

### 2. Chose existing site

There are many existing social platforms that could be used for piggyback prototyping—for example Twitter, Facebook, or Reddit. The researcher should consider the pros and cons of the different platforms. We recommend choosing a platform with public data and an API. In particular, three types of data shared on these platforms may have value to designers [11]:

**People:** User profiles contain a wealth of information such as demographics, interests, and network data. If it is possible to obtain a social graph, then systems that require this structure could be prototyped. In this case, the prototype might require some sort of massive scale snowball sampling [2] to get friends of friends to participate.

**Things:** The things that people talk about create vast amounts of content about current events, interests, and sentiments. A researcher could build a prototype that centers on topics. There exist off-the-shelf natural language process packages (such as NLTK<sup>4</sup>) and topic models that can provide enough accuracy to develop quite sophisticated tools.

**Places:** There are a number of sites where people broadcast their location (Twitter, Foursquare). These platforms may serve different purposes and motivations for publicly sharing geo-location information may be complex [3, 17, 24]. Understanding these intricacies will help align a prototype with user expectations.

<sup>3</sup> <https://www.duolingo.com>

<sup>4</sup> <http://www.nltk.org/>

### 3. Gauge critical mass

Finding critical mass on the chosen site for the target population is key. Critical mass is not an absolute value, but rather is specific to each project. If the goal is to match people based on topics, then enough people need to be talking about a given topic. We suggest gauging critical mass on the chosen site by testing the participant recruitment method without a social component. This pilot can be a simple message sent to users, or it may be a survey. This can help determine characteristics of the target population such as: demographics, behavioral patterns of the target population, and technical aspects such as the types of devices typically used by the participants.

As described in Figure 2, gauging critical mass is an iterative process. Using our experience as an example, we tried in previous work to recruit participants for a study about political discussions on Facebook. We initially planned on recruiting participants who shared a link to a political petition in a public Facebook post. We quickly observed hostility towards our requests via Facebook, which was not the case when we reached out on Twitter about the same topics. The *norms and expectations* were different, and we learned that Twitter was a better choice for our study. This example does not suggest that Facebook is not a viable site for piggyback prototyping. It may be just right for certain systems.

### 4. Build prototype on site

Once the site is chosen and proves to have enough potential participants for a viable study, the social aspects of the prototype can be built. This requires building a messaging infrastructure to reach out to the target population. A researcher could manually message participants, but this might not scale. Because of this, piggyback prototypes will generally be semi-autonomous: running code may find users and send them a template message, yet the researcher will still read every incoming response. Some prototypes might suggest that participants communicate with each other through the existing site. In the collaborative translation example from Figure 2, groups would communicate through Twitter replies. Other prototypes might ask participants to communicate through a low-fidelity interface. For example, Facebook users might be asked to collaborate with others on a simple and easily deployed message board.

There are two components that are well suited to be tested through a piggyback prototype:

*Algorithms:* piggyback prototyping can test social algorithms. Recommender algorithms and natural language processing are examples of such algorithms that are commonly used in social computing and that are particularly difficult to evaluate without critical mass. Since a prototype is rough, these tools need not be perfect. In fact, piggyback prototyping could help evaluate them in a live setting.

*Interaction:* this concerns the messages that are sent to users, the tools available to users through the prototype, and the means that users have to communicate with each other. Piggyback prototyping can help iterate on these aspects.

### 5. Deploy and manage prototype

Next, the researcher or designer deploys their prototype on the chosen site. Systems prototyped with this technique should strictly adhere to Internal Review Board (IRB) processes or corporate ethics boards if applicable, and *always* to ethical standards. Prototypes should not violate privacy and should only present minimal risk.

#### *Obtaining consent*

We worked closely with the IRB at our institution to conduct our study using piggyback prototyping (presented next). We highly recommend doing so. When we deployed our prototype, we obtained a *waiver of consent without need of documentation*. Our participants were aware that they were participating in a study, however, and they had an explanation of the study. Piggyback prototypes can work with documentation of consent as well, as long as the necessary critical mass of participants signs it. Full-disclosure on the research goals and obtaining consent may be prohibitive for some projects. Those studies fall outside the scope of piggyback prototyping.

#### *The role of the researcher and self-presentation*

Even though piggyback prototyping involves running code, it is not completely autonomous: the designer/researcher is still a central part of the prototype. They must present themselves as such to the participants with whom they interact. Moreover, similar to a wizard-of-oz prototype in which the researcher is part of the system, in piggyback prototyping the researcher must be deeply involved in the process. This means that the researcher needs to be available to conduct duties such as answer specific participant questions if appropriate, or to remove participants who have asked to be excluded or who behaved badly.

### 6. Collect metrics and feedback

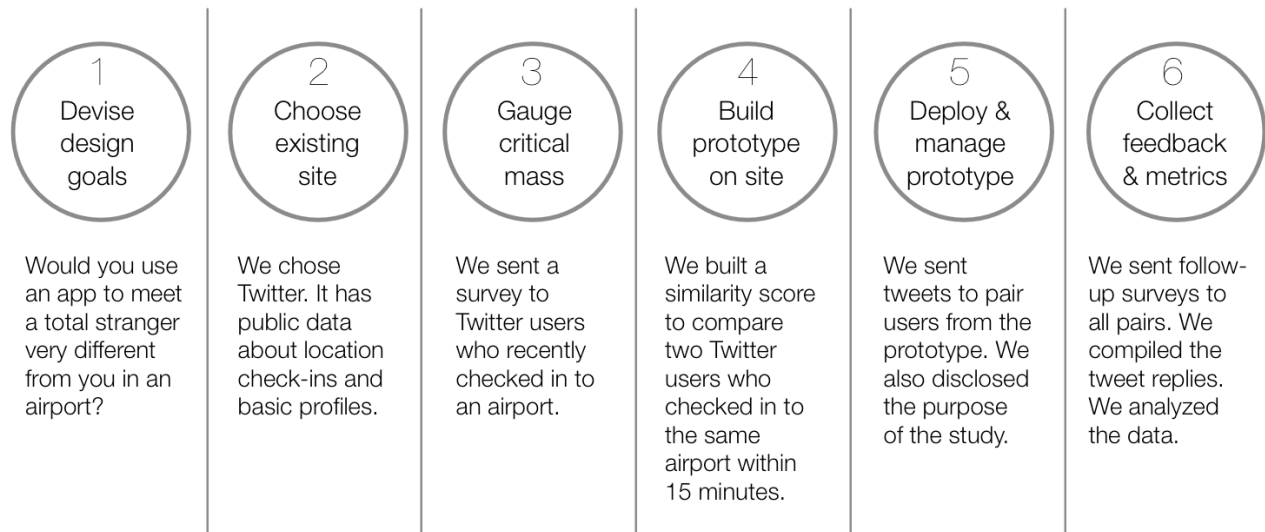
The goal of prototyping in HCI is to evaluate a system in order to iterate on the design [1]. Participants of a piggyback prototype can be sent a follow-up survey to ask about their experience. They might even be interviewed, though the number of participants in this prototyping technique might get overwhelming. We propose that the metrics that can be obtained through piggyback prototyping are:

#### *Engagement metrics*

These are the data that can be obtained from the social platform and supporting ecosystem. For example, the number of clicks on the supporting documentation can serve as one indication of how many people saw the message.

#### *Survey evaluations*

The gauging critical mass survey and final survey are two entry points to ask users for their thoughts on the system. These surveys could ask usability questions about their interactions, or could also ask behavioral questions.



**Figure 3. The 6 stages of piggyback prototyping used in our instantiation. We chose Twitter as our existing site and built a social matching algorithm that pairs users checked-in to the same airport, at the same time, who are different from one another.**

### EXAMPLES AMENABLE TO PIGGYBACK PROTOTYPES

We next present three short examples of existing systems that we believe could have been prototyped with piggyback prototyping, yielding informative research directions. These examples are commercial systems that have already proven their success by being viable commercial products. While we now know how users interact via these sites, it is an interesting thought exercise to apply piggyback prototyping to them.

#### Online dating: Can we prototype OkCupid?

Piggyback prototyping could help evaluate certain components of an online dating site such as the matching algorithm. What algorithm results in the most relationships formed? Who are people looking to meet? For example we could find people who tweet about being single, and about a sports team. Do those who root for the same teams end up getting along? The evaluation could ask whether they would be interested in seeing that person again.

#### Expert knowledge systems: Can we prototype Quora?

We could imagine studying a knowledge system across many people. For example we could find people interested in the same topic in subreddits and ask them to contribute to a shared document. Studying the design of this system has many important research contributions. Who contributes? What topics make the most sense for this? What incentives foster the best answers?

#### Co-location meet-ups: Can we prototype Foursquare?

People go about their daily lives in public places where they are in the presence of others with similar patterns, known as *familiar strangers* [19]. We could try to increase the social capital present in a city by pairing people who tend to tweet from the same location. How should these meetings occur? Does this indeed increase social capital?

### OUR INSTANTIATION OF PIGGYBACK PROTOTYPING

We present an example of deploying the piggyback prototyping technique in a social matching context (see Figure 3). Before building an app, we wanted to explore the conditions under which people would meet face-to-face.

#### 1. Our prototype goal: Will people meet strangers?

The goal of our prototype was to nudge people to meet others different from them in order to combat our natural tendency towards associating with people similar to us, called *homophily* [23]. Previous research on Political Blend, a system designed to introduce people of different political beliefs over coffee [5], showed indications that such meetings could be valuable. Yet, prototyping the system itself was challenging. Through piggyback prototyping, we can answer the following questions:

- Are people willing to meet strangers when prompted through social media?
- Are people willing to meet strangers despite not having much in common?
- What are the design considerations for a system that breaks homophily? What are the pitfalls?

#### 2. Our selected existing social network: Twitter

We chose Twitter as an existing large social network. The Twitter API provided us with the ability to obtain public data including user location, social networks, and profile data. We chose U.S. airports as locations for introducing people since they have benefits such as: many diverse individuals might be collocated; engaging in a conversation with a stranger may be a pleasant way to pass time; and importantly airports have significant security procedures that may lower the risk of meeting a stranger. Furthermore, Airport check-ins on Twitter seemed like a viable route following the success of the TSA Tracker system which asks Twitter users for updates about security lines [21].



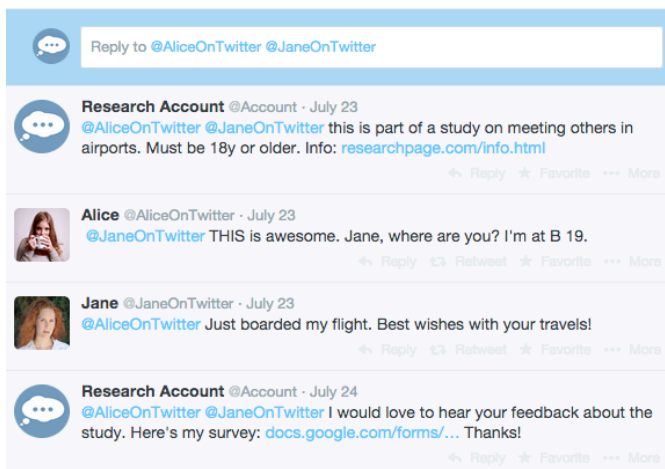


Figure 4. Screenshot of piggyback prototyping in action. Example replies on Twitter, the messages are actual replies from participants. All accounts, names, images, and times have been anonymized.

### 3. Did we find critical mass for airport check-ins? Yes.

We first conducted a formative survey for three purposes: 1) to determine whether people would be willing to meet strangers in airports; 2) to help us determine an expected response rate using airport check-ins on Twitter; and 3) to give us insights into the demographics of this population. The survey asked whether they had met a stranger today in the airport, whether they would use an app to meet strangers in the airport, as well as demographic questions to explore the diversity of the population.

We sent surveys to 1,512 Twitter users who checked in to a U.S. airport between December 2013 and Jan 2014. They received a request to fill out a survey about an app to introduce people in airports. We obtained 213 responses with a response rate of about 14%. From their responses, we found that our target population was interested in meeting fellow travelers and that there was critical mass for users who checked-in within 15 minutes of each other. This was encouraging to pursue further down this road.

### 4. Our prototype algorithm

Once we had early evidence that some significant groups of people had interest in meeting strangers in airports and that it would be feasible, we prototyped the actual interaction involved with matching-users and prompting them to meet-up through Twitter.

	Count	Rate
Visitors on info page	712	11.0 %
Tweets that got replies	576	9.0 %
Survey responses	183	3.0 %
Tweets favorited	61	1.0 %
Replies with location	31	0.5 %
Participants who met	8	0.1 %
<b>Total</b>	<b>6,322</b>	<b>100%</b>

Table 1. Engagement metrics for the social deployment. We paired 6,322 twitter users (which corresponds to 3,161 pairs formed).

For this prototype, we paired users based on their similarities and differences. We built a Twitter similarity classifier that is based on common known dimensions of homophily [4]: content of a user's tweets, their followers, and who they follow. We assigned our participant pairs to three user groups (high similarity, some similarity, low similarity) based on the obtained similarity score. These thresholds were obtained from formative data collection resulting from determining critical mass.

The implementation of this algorithm was done in Python with the Tweepy<sup>5</sup> module to connect to the Twitter API<sup>6</sup>. Through the Twitter Search API we searched for users who checked in to U.S. airports, then for each user we obtained their social graph and their last 20 tweets. Then, we created clusters of users per airport who checked in within the last 15 minutes. Within each cluster we computed the similarity score for each possible pair. We sent a matching tweet to those who were 1) most different, 2) most similar, and 3) finally if some pairs were left they were paired up. Our prototype consisted of approximately 1,000 lines of code, of which a significant proportion was the similarity algorithm.

### 5. Prototype deployment

We ran our prototype on weekdays from May to September 2014. We did not run it continuously during that period since we did not know what to expect from user responses or from Twitter – our account could have been blocked. We started by sending tweets manually, and eventually turned to a semi-automatic system when we found that most responses were positive. We stopped contacting those who requested it (such as not sending them our survey). This only concerned 10 pairs of the 3,161. All the follow-up surveys were sent manually. By monitoring our study closely, we could iterate on some aspects of our prototype:

First, we initially contacted participants as the primary author. This put the spotlight on the researcher's account since it was contacting thousands of users. Instead, we changed to a more general research account. Our contact information was still available in the study documentation.

<sup>5</sup> <http://www.tweepy.org/>

<sup>6</sup> <https://dev.twitter.com/>

Second, we iterated on the message crafted for introductions [25]. Each pair (e.g. A and B) was sent the following prompt: “@A and @B you're both at CLT. Why don't you meet before your planes take-off?” The prompt was followed by a tweet containing the age limit of 18 for participation, and another tweet with information about the study. We got feedback that the tweet about having to be 18 years old was deterring so we included it in the same tweet as the information link. This streamlined the process by only sending two tweets (one to initiate the match and one with the study information).

## 6. Evaluation: survey instruments and other data

The day after we paired Twitter users, we sent them a link to a survey to ask them about their experience with the meet-up (see Figure 3 and Figure 4). In this survey, we asked them whether they did meet up or not. If they did meet up, we asked them questions on these three topics: inter-personal likeability [12], self-disclosure, and follow-up connections. Finally, we asked them how long the meet up lasted. In the cases where the participants were not able to meet up, they were asked why. We asked all participants if they would like to be matched again. Most of the questions were Likert scale.

In addition to the surveys, we also obtained data for our prototype through page views analytics and tweets we received back from our participants. This data was simply obtained from a Google Analytics script inserted on the documentation page residing on our lab server.

## LESSONS LEARNED FROM OUR PROTOTYPE

The goal of our prototype was to see if people would meet face-to-face, and to gain design insight into a system that would prompt people to do so. We did see people meet and the engagement we got with the prototype was enough for us to develop insights into the design of a system in this context. The metrics are summarized in Table 1.

### People were willing to meet strangers in airports

We sent a survey to 1,512 Twitter users who checked in to a U.S. airport between Dec. 2013 and Jan. 2014 and we obtained 213 responses. In this survey, we asked about whether they have talked to a stranger since they have been in the airport. Over half of the users who checked-in to an airport on Twitter had engaged in a conversation at the airport with someone they did not already know (56%). When asked if they would be interested in meeting strangers while they waited, 71 participants (32%) said “yes” and 112 participants (51%) said “maybe.” Only 33 participants (15%) said that they would not be interested in meeting strangers. Of those who said they would not be interested in meeting someone while they waited, there were only 10 of them (21%) who would not use a social app in the airport. Others might install an app to get a coupon with someone while they wait (35%) or to play a social game (20%). These were key findings that allowed us to move forward.

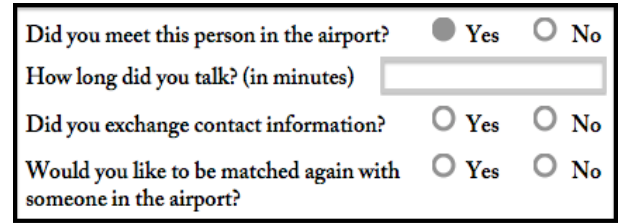


Figure 3. Survey questions for participants who did meet in the airport. They were also asked questions about interpersonal likeability and self-disclosure (not pictured here).

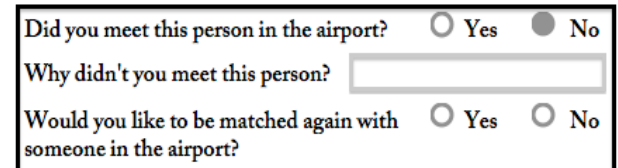


Figure 4. Survey questions for participants who did not meet.

### Airport check-ins on Twitter had critical mass

Since December and January tend to be high travel seasons, we compared the critical mass from this period with a relatively low travel season (end of March, beginning of April) we found that most 15 minute time spans had at least two users checked in to a large airport in the U.S. We came to this finding after some trial and error since we did not know what to expect of people's airport check-in behavior. Since we wanted to pair two people, we determined that 15 minute windows was enough for critical mass of users to conduct our study.

### Participants engaged with our prototype

From May to September 2014, we paired up 6,322 Twitter users (3,161 pairs) who had checked in to airports on Twitter and sent a follow-up survey the next day. We obtained 186 survey responses, of which 182 respondents had not met their match and 4 had met their match. We got 576 Twitter replies and we had 712 unique visitors to our study information page. Our pairing tweets were favorited 61 times. Of the replies we received, 31 had location or contact information. These data suggest that a rough social prototype like the one we deployed can lead to significant amounts of data that help gain insights on the intended interaction. The replied tweets can be analyzed to understand what is happening. The fact that the tweet was repeatedly favorited (61 times) is encouraging. And we obtained enough survey responses to gain a more in depth understanding, which we will talk about next.

An example exchange between participants is illustrated in Figure 4. Other tweet replies we received were:

*“would've made my afternoon”*

*“haha safe travels! Hope you're not #theOnethatGotAway”*

*“I hope you're doing this for awhile it's such a cool idea!!”*

*“I would have participated had I not been so preoccupied with getting my luggage and search for tacos. Next time”*

### **People actually met when prompted by our prototype**

The day after participants were paired, we sent them a follow-up survey to ask about whether they met the other person or not, and what the meet up was like.

#### *Missed connections*

In the deployment, we saw a number of missed connections. 31 participants tweeted their gate or location information as a follow-up to our pairing tweet. We found that the reasons meetings did not occur were: participants saw our tweet too late; the participants were too far away from each other; the participants did not have enough time before their flight; social reason (such as traveling with a family member); some checked in to the airport even though they were not travelling; and finally some had arrived to the airport rather than waiting to depart.

#### *Actual meetings*

What is remarkable is that people actually did meet when prompted through our prototype. In total eight participants met thanks to our prototype. These meetings occurred between people who were highly different (according to our computed similarity score). Of the four reported meet-ups, the participants felt rather neutral about whether they could be friends with the person. This could be simply because a friendship needs more time to develop. Yet, most participants would be interested in being paired again (3 of the 4 pairs). Most meetings exchanged contact information (3 of the 4). One of the pairs, in fact one of the most different pairs, talked about topics that tend to be more controversial, such as religion and politics. In all cases, they talked about their jobs. Family and relationships, and general interests such as music and movies were also talked about in three of the meet-ups. The participants also reported that the meetings lasted 60 minutes in two cases, 30 minutes for one and 5 minutes for another.

### **Future design considerations**

The goal of prototyping is to gain design insight about what would happen when a full system is built and deployed. From our instantiation of the prototype, we learned:

#### *Opt-in system*

Most missed connections might not happen if users initially signed up for the system and were thus expecting these prompts. The challenge with building an opt-in system is obtaining critical mass, this was the reason we did not go that route in the first place. Now that we have an idea that people are interested in meeting others while waiting in airports, we can build a complete system with more confidence about our design decisions.

In an opt-in system, we would have greater control in creating pairs that can meet. For example, we could take into account how much time travelers have until they board, and whether they are arriving or departing. Through this prototype, we were able to determine some of these issues that could be explicitly designed for in our next iteration.

#### *Matching message*

It was surprising to us that participants did not seem to need to know much about the other person to be willing to meet. While it might have been more motivating to know some common interests with the match, we did not see the lack of information about the match to be a large barrier. Perhaps a finding here is that people do not really care to look at the profile of the other person and are actually generally willing to meet a stranger no matter who that person is. A controlled comparative study using piggyback prototyping could more fully explore this. A priori, we thought the introductory message would be key, but it did not seem to matter much in our prototype.

#### *Technical considerations*

Finally, 66% of survey respondents used iOS devices and 24% used Android. This gives us an indication of how to prioritize the development for a future app in this space.

### **Limitations of this specific prototype**

People have different attitudes towards meeting strangers and towards seeking diversity, some are more introverted and some are more extroverted [20]. We did not include a personality survey though it is likely that we got an extrovert crowd that was more willing to engage with a diverse group of people since we only got people who were interested in sharing their location publically. Furthermore, the computed similarity score was approximated from the data we can obtain from Twitter, which is noisy. This might be the case no matter what data is used to obtain a measure of homophily. However, a system in which we could ask for specific profile data may be more accurate.

### **CONSIDERATIONS WITH PIGGYBACK PROTOTYPING**

As a prototype technique, a piggyback prototype is not meant to be a fully completed system. Rather it is rough and flexible: it should be easy to iterate on. As we described, the piggyback prototyping technique is a 6-stage process that provides a scaffolding mechanism for an iterative process for designing large-scale social computing systems. In our instantiation of piggyback prototyping, we learned about ideas that would improve our initial system like having it be opt-in for pairing people according to more fine-grained information. We hope to have shown how other researchers can also implement this approach.

#### **Critical mass**

Obtaining critical mass in any system is extremely complex and not well understood. Users might come because of good design, a well-timed product launch, or simply because of good luck. In our prototype, we knew from our formative survey and data collection that there were enough people checked-in at the same airport at the same time to pair them up. This step is necessary to make sure that a prototype will have enough users.

It is not because a prototype is successful that the resulting completed system will obtain critical mass. However, through this technique we hope to give researchers and de-



signers more tools to consider projects that they might not have had resources to even start otherwise.

### **Volume of users**

Piggyback prototyping concerns large numbers of users. This is unique to this prototyping technique compared to others used in HCI. As such, the evaluation of a piggyback prototype must be catered to this volume. We would argue that a quantifiable survey is more manageable than user interviews. This also means that the resources to manage the volume of participants must be considered. Participants may want information about the study or may personally message the researcher. While we only had two cases of participants emailing us for more information, we can imagine that this could quickly become difficult to manage if every participant had emailed us.

### **Choosing appropriate metrics**

Our piggyback prototype made us reconsider the traditional evaluation metrics of social computing systems. We had many participants due to the fact that Twitter had many people checking in to airports, thus the fact that we had 6,322 users does not speak to the merit of our system. What does? We looked more deeply at survey results, engagement metrics and user responses to get a sense for the value of our system. Similarly, researchers who employ piggyback prototyping should determine for their project what metrics and feedback they would like to obtain.

What was important to us was to determine whether some people would meet up and whether those meet-ups were meaningful. Some might consider the four meet-ups we saw to be a limiting aspect of our study: “four is a small number, so the impact of the system is underwhelming.” Yet, we saw that those meet-ups were highly successful from the survey responses despite the fact that the people paired up were highly different. This finding is surprising and significant enough to continue down this line of work.

### **Longitudinal studies using piggyback prototyping**

There is a dilemma around the longitudinal aspect of piggyback prototyping. On the one hand, a script could constantly run to obtain data over a long period of time, as long as one does not get blocked from crawling the site. (That is, a site could interpret high levels of activity against it as an attack and shut the script down.) At the same time, one must consider possible user fatigue. To our knowledge, Twitter users who tag their location are not constantly bombarded with research requests. While this study shows that at the time of the study, a significant number of people welcomed our intervention, this could also be due to some novelty effect. If these requests were a more frequent occurrence, Twitter user behaviors may change. While we see promise in the feasibility of this technique, we are also aware that an over-abundance of piggyback prototyping might drastically change behavior, and therefore the feasibility of the technique. This kind of reflexivity is present in most social systems.

### **Generalizing outside of Twitter**

Our piggyback prototype was deployed on Twitter. This platform was ideal at the time of this study because it contained a large public dataset of location check-ins through its tight integration with geo-locating services such as Foursquare. We believe this platform could work for many other types of piggyback prototypes. Though we imagine that other platforms may be just as suitable. Facebook and Reddit are examples of platforms on which users can message each other, and thus provide an infrastructure for piggyback prototyping. Certain limitations (such as the current \$1 cost to message a non-friend on Facebook) should be considered. Each project should consider the implications of the chosen existing site. If Twitter is widely popular and accessible today, it could be different tomorrow. Piggyback prototyping would still be feasible, but a careful understanding of available social platforms is necessary.

### **What falls outside the scope of piggyback prototyping?**

Not all large-scale social computing systems can be prototyped with piggyback prototyping. Three types of projects may not be well-suited to this technique: 1) those that deal with sensitive or protected data, 2) those that cannot disclose the purpose of the study to the user, and 3) those that require anonymity. For example, if the researcher has access to private data like direct messages on Twitter, then that data should not be shared with other users. Or, if the system depends on anonymity, then leveraging existing non-anonymous social networks might make it difficult to evaluate in situ. Considerations for privacy are especially important and not always straightforward. For example, we suggested that piggyback prototyping could test social algorithms such as matching algorithms. However, some algorithms might reveal information from public data that most users would not have been able to find.

### **Biases and limitations**

People who publicly share broadcast messages are a self-selected group. For example, they might be more extroverted or more narcissistic. Beyond how this might impact findings in our own study on location sharing, this bias must also be considered in most piggyback prototyping systems. Second, using certain sites may not be accessible to all researchers. For our study, we used a Twitter account that was first a personal account and then evolved into a study account. As such, Twitter’s automated defenses did not block it. It is possible that an account specifically created for a piggyback prototype might exhibit behaviors that would get it blocked.

### **Towards a social toolkit**

In HCI, a basic building block of software UI prototyping was the development of UI toolkits that contained modular pre-defined UI components that could quickly be assembled. Could we consider the social computing systems counterpart? If we compare piggyback prototyping to the Model-View-Controller paradigm, we could use existing social data as the Model and we prototype the View and the Controller parts to varying degrees. In our example of pair-

ing users who checked in to Twitter, we emphasized a prototype of the Controller. Others might choose to focus more on the View to prototype the visual aspect of the system. For example, Groupkit [9] provides a toolkit for video-conferencing, which facilitates the development of critical components (such as sessions) for these types of systems. Similarly, a toolkit for large social systems could be envisioned as follow-up to piggyback prototyping.

## CONCLUSION

We developed *piggyback prototyping*, a prototyping technique for large-scale social computing systems. We described an example of using this prototyping technique to pair people checked-in to airports. The goal of this project was to see whether people would meet despite differences, yet we quickly realized that we needed a critical mass of users for the system to be useful. We took advantage of pre-existing critical mass on Twitter and deployed a prototype of our matching algorithm on top of Twitter. Through this we learned that travelers who shared their location on Twitter responded positively to social matching prompts. We were surprised that many were willing to meet in the context of our prototype. We found that piggyback prototyping addressed the shortcomings of HCI prototyping techniques when it comes to large-scale social computing systems: piggyback prototyping allowed us focus on *what people do on a social computing system* rather than *how to attract people to the system*.

## REFERENCES

1. Beaudouin-Lafon, M. and Mackay, W. Prototyping Tools and Techniques. *The Human-Computer Interaction Handbook*. 2002. p. 1006-1031.
2. Biernacki, P. and Waldorf, D. Snowball Sampling: Problems and Techniques of Chain Referral Sampling. *Sociological Methods & Research*. 1981.10(2):141-163.
3. Cramer, H., Rost, M. and Holmquist, L.E. Performing a check-in: emerging practices, norms and 'conflicts' in location-sharing using foursquare. *MobileHCI*. 2011. p. 57-66.
4. DeChoudhury, M. Tie formation on Twitter: Homophily and structure of egocentric networks. *SocialCom: IEEE Conf. on Social Computing*. 2011.
5. Doris-Down, A., Versee, H. and Gilbert, E. Political blend: an application designed to bring people together based on political differences. *Communities and Technologies*. 2013. p. 120-130.
6. Foner, L. and Crabtree, I. Multi-Agent Matchmaking. *Software Agents and Soft Computing: Towards Enhancing Machine intelligence, Concepts and Applications*. 1997. p. 100-115.
7. Gaver, B., Dunne, T. and Pacenti, E. Design: Cultural probes. *Interactions*. 1999. 6(1): p. 21-29.
8. Gilbert, E. Designing social translucence over social networks. *CHI*. 2012. p. 2731-2740.
9. Greenberg, S. and Roseman, M. Groupware Toolkits for Synchronous Work. *Computer-Supported Cooperative Work: Trends in Software* 7. 1999.
10. Grudin, J. Groupware and social dynamics: eight challenges for developers. *Communications of the ACM*. 1994. 37(1): p. 92-105.
11. Guy, I., Jacovi, M., Perer, A., Ronen, I., Uziel, E. Same places, same things, same people? mining user similarity on social media. *CSCW*. 2010. p. 41-50.
12. Hancock, J.T., Toma, C.L. and Fenner, K. I Know Something You Don't: The Use of Asymmetric Personal Information for Interpersonal Advantage. *CSCW*. 2008. p. 413-416.
13. Houde, S. and Hill, C. What do Prototypes Prototype? *Handbook of Human-Computer Interaction Second Edition*. 1997. p. 367-380.
14. Iachello, G., Truong, K.N., Abowd, G.D., Hayes, G.R., Stevens, M. Prototyping and sampling experience to evaluate ubiquitous computing privacy in the real world. *CHI*. 2006. p. 1009-1018.
15. Irani, L. and Silberman, S. Turkopticon: Interrupting Worker Invisibility in Amazon Mechanical Turk. *CHI*. 2013. p. 611-620.
16. Lim, Y-K., Stolterman, E. and Tenenberg, J. The Anatomy of Prototypes: Prototypes as Filters, Prototypes as Manifestations of Design Ideas. *ACM Trans. on Computer-Human Interaction*. 2008. 15(2).
17. Lindqvist, J., Cranshaw, J., Wiese, J., Hong, J., Zimmerman, J. I'm the mayor of my house: examining why people use foursquare - a social-driven location sharing application. *CHI*. 2011. p. 2409-2418
18. McDonald, D., Gokhman, S. and Zachry, M. Building for social translucence: a domain analysis and prototype system. *CSCW*. 2012. p. 637-646.
19. Milgram, S., in *The Familiar Stranger: An Aspect of Urban Anonymity*. 1972.
20. Munson, S. and Resnick, P. Presenting Diverse Political Opinions: How and How Much. *CHI*. 2010. p. 1457-1466.
21. Nichols, J. and Kang, J-H. Asking questions of targeted strangers on social networks. *CSCW*. 2012. p. 999-1002
22. Rettig, M. Prototyping for tiny fingers. *Communications of the ACM*. 1994. p. 21-27.
23. Smith, J., McPherson, M. and Smith-Lovin, L. Social Distance in the United States: Sex, Race, Religion, Age, and Education Homophily among Confidants, 1985 to 2004. *Annual Review of Sociology*. 2014. p. 432-456.
24. Tang, K.P., Lin, J., Hong, J., Siewiorek, D. and Sadeh, N. Rethinking location sharing: exploring the implications of social-driven vs. purpose-driven location sharing. *Ubicomp*. 2010. p. 85-94.
25. Terveen, L. and McDonald, D.W. Social matching: A framework and research agenda. *ACM Trans. Computer-Human Interaction*. 2005. p. 401-434.