



# Transporter

A Real-Time Public Transit App  
Designed for Bay Area Riders

## ABSTRACT

Transporter is a novel public transit application for iPhone, built to help riders get around the Bay Area. Its design is the product of user research into how people use public transportation and the difficulties they encounter. For example, you might be standing at a stop and want to know when the next bus is coming. All it takes is two taps to find that out. Once you're on your way, it only takes a tap to see all the upcoming stops and when you'll arrive at any of them. For trips with transfers, Transporter features a real-time trip planner, which uses real-time arrivals predictions to give you an up-to-the-minute travel plan that updates as you go. Miss a transfer? Transporter will recalculate the rest of your trip. These are just a few of the many thoughtful features that make Transporter the best public transit app for the Bay Area.

## TEAM

Transporter was designed and developed at the UC Berkeley School of Information by Ljuba Miljkovic for his Master's thesis and by Thejo Kote for Information 213, a course in user Interface design. Ljuba built the Transporter iPhone app and lead the interaction and visual design of the project. Thejo built Transporter Server (described later) and assisted in the design and testing of the app.

## ORIGINS

Like so many design projects, the impetus for Transporter was a personal need. As a frequent Bay Area transit rider, I was unsatisfied with the state of transit apps available for the iPhone. The Maps app included with the iPhone relies on transit schedules, so it is useful for planning trips only to the extent that the buses and trains run on time, which is often not the case. Services like NextBus<sup>1</sup> and the BART real-time service<sup>2</sup> – which track vehicle GPS locations and predict arrival times – attempt to ameliorate the problem. Indeed, apps like iCommute<sup>3</sup> and Routesy<sup>4</sup> are built around these services, however the data they provide is limited to arrivals predictions for a given stop (e.g. the next 11 bus comes in 9 minutes). It seemed to me that more could be done with the available data, like building a trip planner that uses real-time arrivals data instead of schedules. Until very recently, this had never been done (at least not for Bay Area transit systems)<sup>5</sup>. Moreover, it seemed to me that the current crop of apps did not reflect the broad needs of public transit riders for the Bay Area. I decided then that the goal of my Master's project would be to build an app that does.

## CONCEPT

The first step in deciding what kind of application to build was to better understand who we'd be building it for. Thus began a series of semiformal interviews with transit riders from around the Bay Area. Over the telephone or in person, we asked them to talk about how they use public transit, what troubles they encounter, and what tools they use. After about a dozen or so of these interviews, patterns of usage began to emerge. For example, we discovered (not surprisingly) that people don't like waiting at a stop. Further, they hate arriving at a platform or bus stop and realizing that they've just missed the train or bus.

After reviewing our notes from these interviews, we discovered that riding public transit often produces a generalized sense of anxiety in riders – rooted perhaps in one's loss of control, which is no doubt worsened when transit vehicles do not arrive as scheduled. One interviewer described the following scenario: he's at a party and wants to know when he should leave so that he can catch a bus and then a train with a minimum of waiting. The challenge in this scenario is that the train runs only once an hour and though it will likely depart on time, the bus may not. Further, tools that can predict when the bus will depart from his location cannot tell him when it will arrive at the train station. His only option is to leave the party early and hope the bus he catches will take him to the station on time for the train. If he doesn't make it, he'll have to wait nearly an hour for the next one.

From these concerns, we developed an *application design definition*: a statement of the product's goals and scope: "A general-purpose transit app for the Bay Area that reduces rider anxiety." At this time, we also agreed on a target user group: Bay Area residents who are at least somewhat familiar with the transit systems. Though this is a fairly broad group, it deliberately excludes visitors and tourists, who we believe would require a different kind of transit app altogether. It was our hope that this design definition would keep us grounded during the ambiguous sketching and prototyping phases that were coming up.

## TRANSIT USE CASES AND CONCERNS

Our interviews also revealed the kinds of transit trips people take as well as the concerns that accompany them. These trips fall broadly into two groups: ad hoc trips and planned trips.

### **Ad hoc Trips**

Ad hoc trips are typically short or familiar transit trips — taking the bus to school, for example. In these cases, riders already know which lines to take and where they stop. However, because buses often do not arrive as scheduled, riders lack the precise arrival times for the lines that they're

interested in. In these circumstances, having real-time arrivals predictions is very useful. When we unpacked this need, we discovered several important user questions:

- When is the bus coming to a particular stop?
- When is the bus coming to the stop I'm currently standing at?

Once on-board, new questions arise:

- Where am I currently?
- When is my destination coming up?

To answer these questions, Transporter would need to include real-time arrivals in a manner not yet seen in other public transit apps.

## **Planned Trips**

Planned trips are typically longer or unfamiliar transit trips - taking transit to a job interview, for example. In these cases, riders may not be familiar with the lines they need or where to catch them. Synthesizing our interview responses produced the following sequence of user questions we wanted Transporter to answer:

- What buses/trains should I take? When?
- How far am I from the next connection?
- What if I miss a connection?
- What if the connection is late? Is there another bus/train I can take instead?

The concerns that accompany planned trips are rooted in the inherent uncertainty of public transit travel. Riders need frequent reassurance that the buses and trains they plan to connect to are on time. And if they're not, they need to know how this will affect their trip and what their alternatives are. From this, it became clear that Transporter needed a real-time trip planner, one that could adapt to the arrival times of upcoming connections.

## **PROTOTYPES**

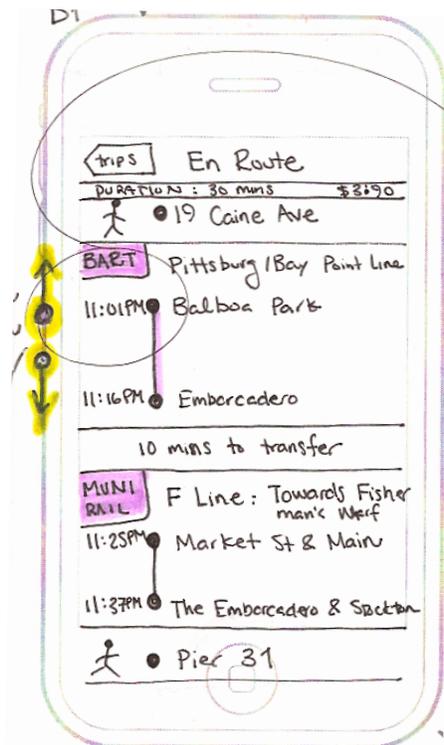
Long before we were ready to develop the application code, we needed to prototype its user interface in lower fidelity media. Broadly, this consisted of two major phases: paper prototyping and interactive prototyping.

## Paper Prototypes

When we first began sketching out possible user interfaces for Transporter, it wasn't very clear how the application should be organized. We began by structuring the application layout by grouping the major functions of the app: planned trips in one tab and ad hoc trips in another.

### PLANNED TRIPS

In the planned trips area, users can enter their start location and destination to receive real-time transit directions. The major challenge of this area was to design a layout for the transit directions that was clearer than directions provided by the Maps app or Google Transit, which I think are needlessly verbose and not as clear as they could be. I propose a layout in which the trip is divided into transit segments, one per bus/train line, starting at the top and moving downward. Transfer times appear as dividers between the segments. (Figure 1)



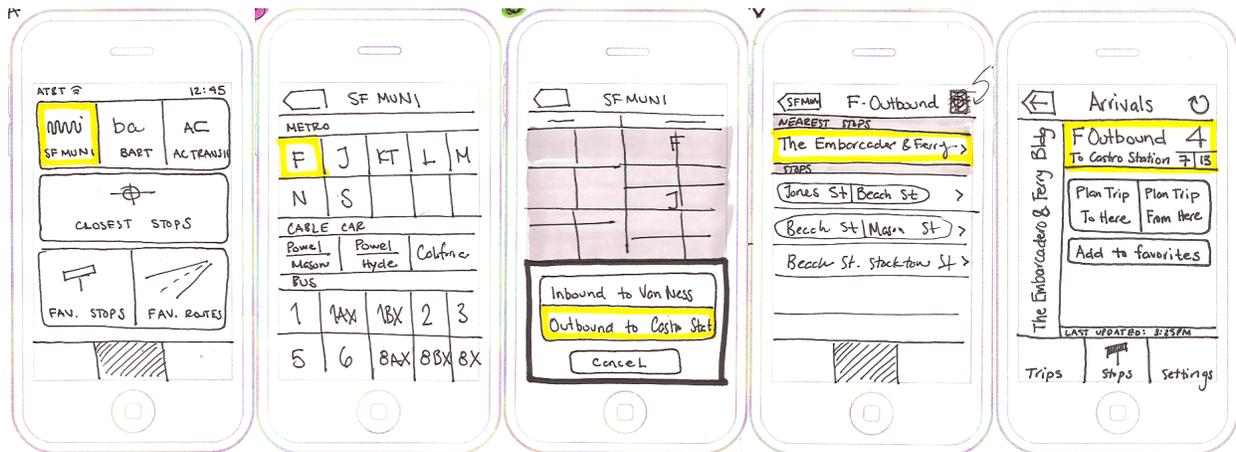
**Figure 1.** Paper prototype of planned trip instructions

This approach favors graphical instructions over written ones. For example, the iPhone Maps app would tell users to “Take BART towards Pittsburg/Bay Point. Departs at 11:01 PM.” and then “Get off BART at Embarcadero BART. Arrives at 11:16 PM.” The proposed layout above illustrates transit segments as lines between two stops with the departure and arrival times to the left of each

stop. The agency and name of the line, in this case BART and Pittsburg/Bay Point, are at the top of the segment. We believe this layout closely reflects people’s model of public transit travel: as a series of transit rides with transfer times between them.

## AD HOC TRIPS

The ad hoc trips area of the application was called "Stops" at this time, and grouped features to find arrival times of any bus or train in the supported agencies. It also supported finding nearby stops, and finding arrival times for "favorite" stops and lines. Two major challenges emerged in this section of the app: navigating to an arbitrary stop and finding nearby stops. To navigate to the arrivals screen for any stop in a supported transit agency, you must select (in order) the agency, line, direction, then stop. For example you might select SF Muni, the F line, headed Outbound, at Embarcadero and Ferry Building. (Figure 2)

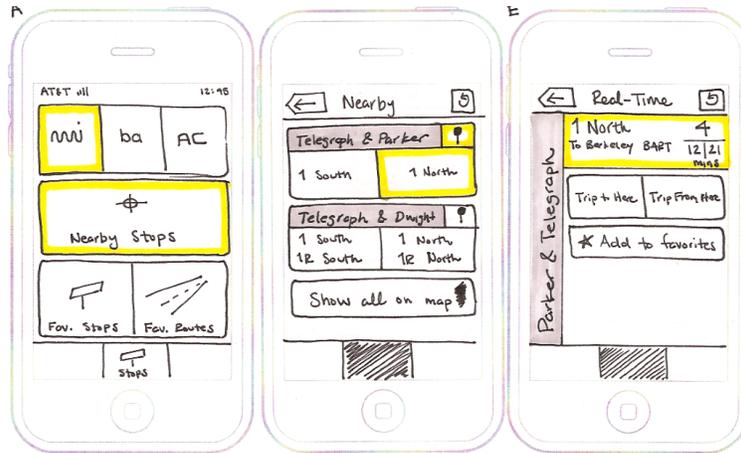


**Figure 2:** User interface flow for selecting a transit line and stop, in this case F Outbound at Embarcadero and Ferry Building.

In this user interface task flow, a user first selects SF Muni from the three supported agencies. Muni metro, cable car, and bus lines are then displayed in a grid to minimize the need for scrolling. When the user selects a line, a modal menu slides up from the bottom prompting her to choose either the inbound or outbound direction. Next, an ordered list of stops for that direction appear, with the closest stop at the top. Once a stop is selected, its real-time arrivals are displayed.

Finding nearby stops is intended for users who are currently standing at a stop and want to know arrival times of the buses/trains that stop there. At this stage of prototyping, our best solution was to list the stop names in order of proximity. To simplify the list somewhat, we grouped the

stops of the same name (which appear on opposite sides of an intersection) so that they wouldn't clutter the list. (Figure 3)

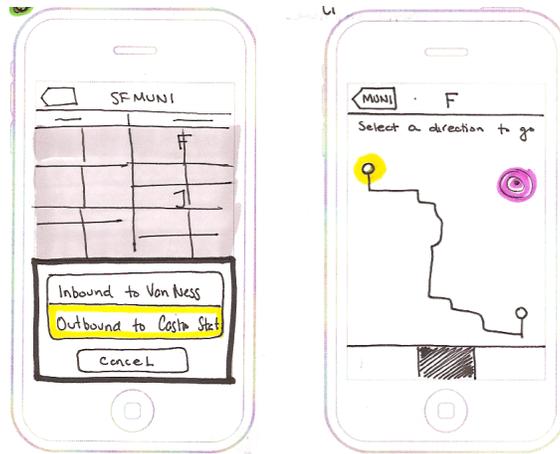


**Figure 3:** User interface flow for selecting a nearby stop.

In this user interface task flow, a user first selects "Nearby Stops" to see a list of nearby transit stops. In this case Telegraph & Parker is the closest. Only the 1 bus stops there, heading north on one side of the street and south on the other. The user selects the northbound stop to see that bus's arrival times.

## USER TESTING

Throughout the paper prototyping process, we presented our sketches to transit riders familiar with user-centered design methodology and obtained their feedback. These sessions were structured such that the user would have to perform tasks currently supported by the paper prototype. We would note and explore any difficulties they encountered to help improve the prototype. For example, we discovered that selecting a direction is the most ambiguous part of navigating to the stop arrivals screen because people are often unfamiliar with direction names. Our user testing revealed that a map-based selection system would provide greater geographic context and make selection easier. (Figure 4)



**Figure 4:** Left, modal name-based direction selection. Right, map-based direction selection (user location in pink).

In the map-based direction screen, a user sees a map of the line, its endpoints, and her current location. To choose a direction, the user need only tap one of the lines end-points. This approach provides a much-needed geographic context and helps users choose a direction more confidently.

Paper prototyping was an invaluable step in the design process. It allowed us to explore application organization possibilities with a minimum investment of time and resources. It was also the most ambiguous and difficult stage of the project. Major questions of how Transporter was going to meet the user needs we identified had to be answered at this stage. From seemingly infinite possibilities, we were forced to choose but a few and embody them on paper. Once we had achieved a design that seemed right and performed well with users, we moved to a higher fidelity prototype.

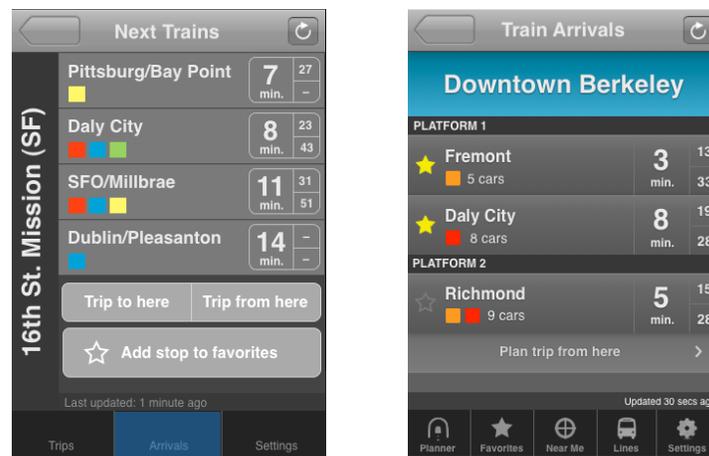
## Interactive Prototypes

Up until this point, user interaction with our prototypes had been simulated by moving pieces of paper around as users tap on illustrated controls. To better understand how users would truly interact with Transporter, we developed interactive prototypes using Apple's Keynote presentation software. This approach offers the advantage of interactivity (users can click on elements in one slide and be taken to another) without the overhead of writing code or application logic. Keynote is a capable vector drawing tool, so we used it to build the interface as well.

In addition to fleshing out the interactive components of the application, this stage of prototyping forced us to grapple with issues of pixel-accurate layout. We could no longer make assumptions about how much content would fit in a given area of the screen. As with the paper

prototype, we periodically tested our designs with users and improved on them based on their feedback. We discovered several problems we did not detect during our earlier prototyping phase, likely because the sketches weren't of high enough fidelity for users to perceive the problems.

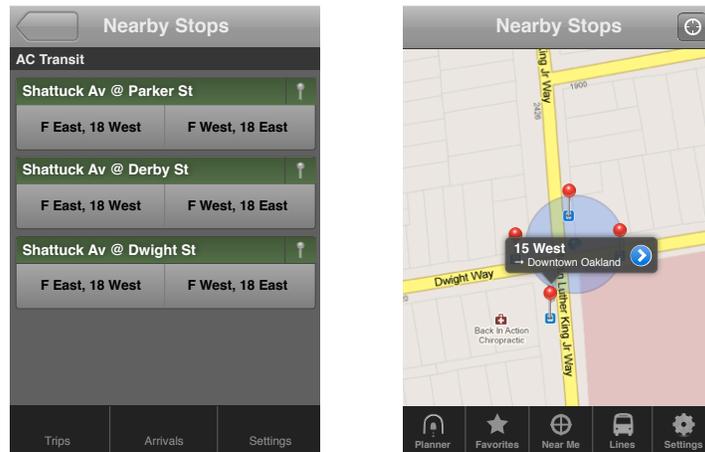
For example, in the paper prototype and early interactive prototypes, the stop name is oriented vertically on the stop screen. This was done to leave as much room for transit lines as possible. During user testing of the interactive prototype, however, it became clear that no one was reading it. People's eyes simply weren't moving to the bottom left corner to read the stop name. (Figure 5)



**Figure 5:** Left, an early interactive prototype of the stop arrivals screen with stop name on its side. Right, final interactive prototype with stop name on top.

User testing also revealed that "Stops" (later, "Arrivals") was a poor navigation tab name. It wasn't clear to users what they would find there. So, we removed one level of hierarchy from the application navigation structure by moving "Favorites" and "Nearby Stops" to the tab bar and renaming the "Stops" tab to "Lines". Now, specific user tasks had their own tabs, making their discovery far more likely. (Figure 5)

Finally, we discovered that our approach to the "Nearby Stops" screen was completely wrong. Most of our users had a hard time picking the correct stop because they didn't know which of the two stops listed under the same name they wanted. We went back to our use cases and realized that this particular feature would only be used if the user were standing at the stop they were interested in. Thus, it makes much more sense to show a map of the user's location and nearby stops. That way, they could choose the one on the side of the street they are standing on. Our follow-up user testing confirmed this. (Figure 6)

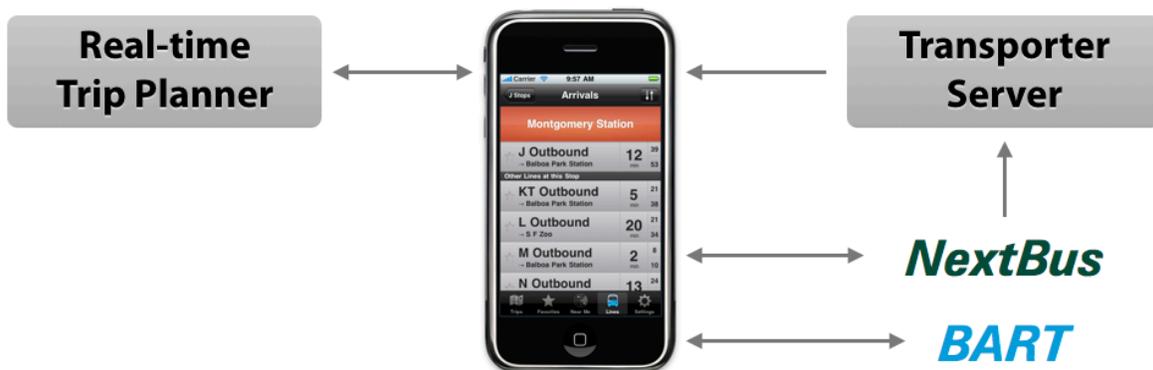


**Figure 6:** Left, an early interactive prototype of the “Nearby Stops” screen, with stops ordered by proximity. Right, final interactive prototype with map-based stop selection.

We knew we had reached the end of this prototyping phase once our user testing stopped revealing problems in the design. At that point, it was time to move to code. Only a working application, one that delivered actual transit data, could reveal any remaining design problems.

## DEVELOPMENT

When people encounter the Transporter app, they'll likely be unaware of the server components that make its features possible. In fact, beyond accessing the real-time BART and NextBus APIs, Transporter relies on two custom server components: one for transit system updates and the other for real-time trip planning. (Figure 7)



**Figure 7:** Transporter system architecture.

## Transporter Server

The transit data that Transporter uses comes from NextBus and BART. This includes stop locations and names, bus and train line information, route paths, etc. This information is very complex and needs to be preprocessed before Transporter can use it. We developed Transporter Server to automatically import this data from the web, process it, and push any transit system changes to the app over the internet. We also use Transporter Server to enhance the data. For example, from one stop arrivals screen you can see the arrivals for the stop across the street by tapping the “switch directions” button in the top-right corner. (Figure 8) Linking these two stops was done on the server.



**Figure 8:** Animation sequence after tapping the “switch directions” button. Tapping this button from the Montgomery Station outbound screen flips the lines to show inbound arrivals on the opposite platform.

## Real-time Trip Planner

When this project began in early 2010, it seemed likely that we would have to build our own real-time trip planner since we knew of none that existed. However, over the course of the semester we discovered a research group at UC Berkeley's Department of Civil and Environmental Engineering that had built one as part of a research project on human transit behavior modification<sup>5</sup>. We negotiated a license agreement that allowed us to use the trip planner API in Transporter in exchange for design consultation for their own project.

## Transporter App

The client iPhone application was built using the Apple Xcode development environment and Cocoa framework. The app contains a database of every stop and line in BART, SF MUNI and AC

Transit that currently supports real-time arrivals. We would have liked to support more agencies, however, these are the only major transit agencies in the Bay Area that provide real-time data.

When the user navigates to a stop screen, real-time arrivals for the lines that serve that stop are requested from either BART or NextBus (which serves AC Transit and SF Muni). When a user requests directions for a trip, the app queries the real-time trip planner. Throughout the trip, Transporter fetches real-time arrivals from NextBus and BART to update the times in the trip plan. Finally, the app checks Transporter Server for service changes and downloads updates over the internet as needed.

## **USABILITY/BETA TESTS**

Before development of Transporter was complete, we conducted a series of final user tests intended to uncover any problems in the implementation of our interactive prototype. The goal was to catch usability problems in the working app that we hadn't been able to encounter at earlier prototyping stages. The tests were divided into two categories: informal beta testing and formal usability testing.

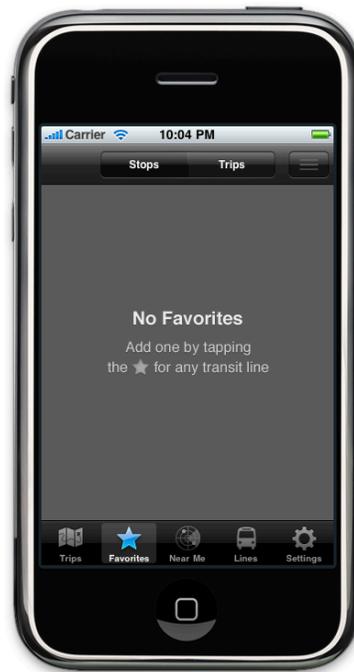
### **Beta Tests**

It is our hope that Transporter will be used by hundreds of people once it's released. To get a broad range of perspectives before launch, we initiated a beta program where interested users could sign up to receive prerelease versions of the app in exchange for their feedback. Users can report on their experiences with the app by tapping on the prominent (and temporary) "Bugs" tab and sending us an e-mail. This kind of testing is intended to discover edge case bugs, stability problems, and other issues that might not come up in a more focused user test. To date we have 20 beta testers who have already identified a number of small bugs previously unknown to us. The beta program will continue as long as Transporter is being developed.

### **Formal Usability Tests**

We conducted a series of usability tests with three users who had never seen Transporter before at any stage of development. We asked them each to perform three specific tasks while we observe their behavior. We noted any problems they encountered, but did not interfere in their completion of the tasks. Afterwards, we interviewed them to determine why they might have been confused at any point during the tests. Compared to beta testing, this approach was more focused on identifying problems specific to first-time Transporter users. For example, when we asked users to save a particular BART line as a favorite, they all tapped on the "Favorites" tab but saw nothing

there and became confused. It became clear that we needed some indication that favorites were accessed from this tab, but added somewhere else. (Figure 9)



**Figure 9:** Initially, the favorites screen was blank. After user testing, we added a message to let users know how to add favorites.

Overall, we were pleased that the problems we encountered during both the beta and formal usability testing were minor. We believe this is due to our thorough testing at prior prototyping stages – by the time we had built the app, all major usability problems had been addressed.

## DEMONSTRATION

How one uses Transporter depends so much on geographic context. Instead of describing how its features address user needs, the following video shows it:

<http://www.youtube.com/watch?v=CNBXwR-HxEU&hd=1>

## CONCLUSION

Over the course of the semester, our singular overarching goal was to design a mobile application that would reduce public transit rider anxiety. We conducted numerous preliminary interviews to learn about people's public transit use and the problems they encounter. Then, through a series of prototypes of increasing fidelity, guided by the feedback we received from users at each stage, we were able to build an iPhone app that addresses these needs. Ultimately, this design project has

been about learning how people use public transit and weaving that knowledge into the design of Transporter.

We encountered many challenges along the way. Though we present the above design process as a linear progression through a series of stages, the reality was far more complex. Each stage of the design process was a challenge, not just to find the "right solution" but to truly understand the underlying problem we were trying to solve. For example, our grasp of transit use cases became much firmer *after* we completed the paper prototype. Some uncertainty about how Transporter would address our users' needs persisted even through part of the development phase. We don't consider this a failing but rather a reality of the design process. Your understanding of users' needs may not become completely clear until you've engaged with the problem at a greater level of detail.

Perhaps the most important lesson we learned over the course of this project is that straightforward design ideas can often be very difficult to implement. This is because when we say designs are straightforward, we mean from the perspective of the user. Creating a simple user interface often means working very hard to manage the necessary complexity of the system behind the scenes. This was the case with complex bus lines in the AC transit system. For example, the 51B bus runs from Rockridge BART station to either the Berkeley Marina or University Ave. & 3rd St. and back. When a user selects that line and is prompted to pick a direction, there are technically four directions to choose from:

1. Rockridge BART to Berkeley Marina
2. Rockridge BART to University Ave. & 3rd St.
3. Berkeley Marina to Rockridge BART
4. University Ave. & 3rd St. to Rockridge BART

We decided to limit the number of directions on this screen to two, in this case options 1 and 3. However, once the user picks the stop they want, they will see the arrival times of any 51B bus direction that stops there, even if it wasn't available to choose from in the direction screen. Someone extremely familiar with the 51B line might be initially confused by this, but most people think about bus lines as having only two directions, so why confuse them unnecessarily? Once they reach a stop screen and see two 51B lines headed to different destinations, they can wait for the one they want. Our current solution is not perfect. We would prefer to somehow group the two 51B lines in the stop screen, but this will have to wait until the next version.

## **FUTURE PLANS**

We hope to publish Transporter to the iPhone App Store in June 2010. It will be free to download, with the option to pay as much as users feel it's worth to them. From there, we hope to continue development of new and innovative features, the ideas for which came from the initial interviews. For example, many of the people we spoke to would like to receive notifications of bus arrivals or of changes in a planned trip schedule. Going forward, our plan is to conduct a miniature iterative design process for every such new feature.

## **SPECIAL THANKS**

We would like to thank Professor Kimiko Ryokai for her guidance and support during this project and Kyle Youngblom for designing the Transporter icon and consulting on the app's overall visual design. We would especially like to thank Jerry Jariyasunant and his team in the department of Civil and Environmental Engineering for developing the real-time trip planner API Transporter uses.

## REFERENCES

<sup>1</sup> [www.nextbus.com](http://www.nextbus.com)

<sup>2</sup> [www.bart.gov/](http://www.bart.gov/)

<sup>3</sup> iCommute SF. <http://itunes.apple.com/us/app/icommute-sf/id310607155?mt=8>

<sup>4</sup> Routesey. <http://itunes.apple.com/us/app/sf-tracker/id355191825?mt=8>

<sup>5</sup> Jariyasunant, Jerald et al., Mobile Transit Trip Planning with Real-Time Data. [www.baytripper.org/10-3748.pdf](http://www.baytripper.org/10-3748.pdf)