Design Analysis - Pick-Me-Up!

Team: VRAM
Rohan Mahajan, Vu Le, Andrew Song, Minh Tue Vo
MIT 6.170, Software Studio

April 28, 2013

1 Overview

PickMeUp is a service that helps medium and large event organizers to reduce traffic congestion by offering alternative transportation modes for their attendants. PickMeUp only allows people to share cab/rides if they are going to the same event, and filter suggestions based on regions and social networks like colleges and Facebook. This web service differs from existing car or cab sharing services by only focusing on events and increasing trustworthiness by several filters, thus presenting an immediate, compelling reason for the event attendees to use the service. Moreover, the service appeals to the event organizers as well in a sense that they can obtain the data on how many people are coming by which transportations, and remove the difficulties of attracting attendees due to distance and transportation problems.

Below are the context diagrams for both the basic and the extended version of the Pick Me Up web application. For the extended version, we introduce

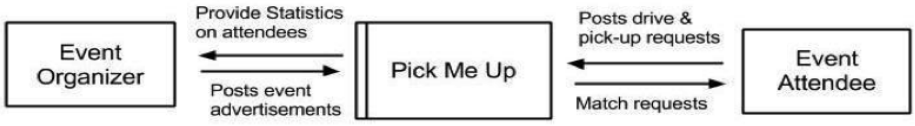

Fig 1.1 Context Diagram for "Pick Me Up" basic version

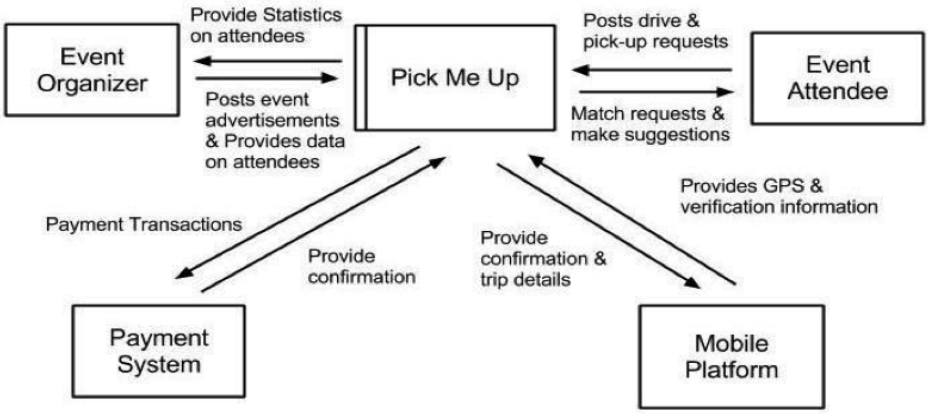

Fig 1.2 Context Diagram for "Pick Me Up" extended version

two new components (payment system, mobile platform) to render the minimum viable product into fully-functioning service product. However, due to the time constraint of the project, these could be left out for the prospective implementation.

2 Concepts

There are three important concepts both for the basic and the extended version: **User**, **Event**, and **PickUp**.

- **User:** represents an event organizer, driver, requestor. The organizer can post new events and gain statistics on how many people are coming through which transportations. The attendant of the events can post a new request - either to share a cab or a car.
- **Event:** represents an event such as a concert or conference. Event has properties such as Name, Description, Location and Date Time. Location is either a pair of GPS coordinates (latitude, longitude) or a physical entity (landmarks such as the Prudential Building).
- **PickUp:** - Pick Up is a request made by either the driver or the requester. It includes the locations and participants. There are two types of PickUp requests: car sharing, which needs to identify one user as a driver, and

cab sharing, which does not require a driver. The system will give suggestions to User based on the input information (such as Users preferences, Facebooks friend list and the regions of residence).

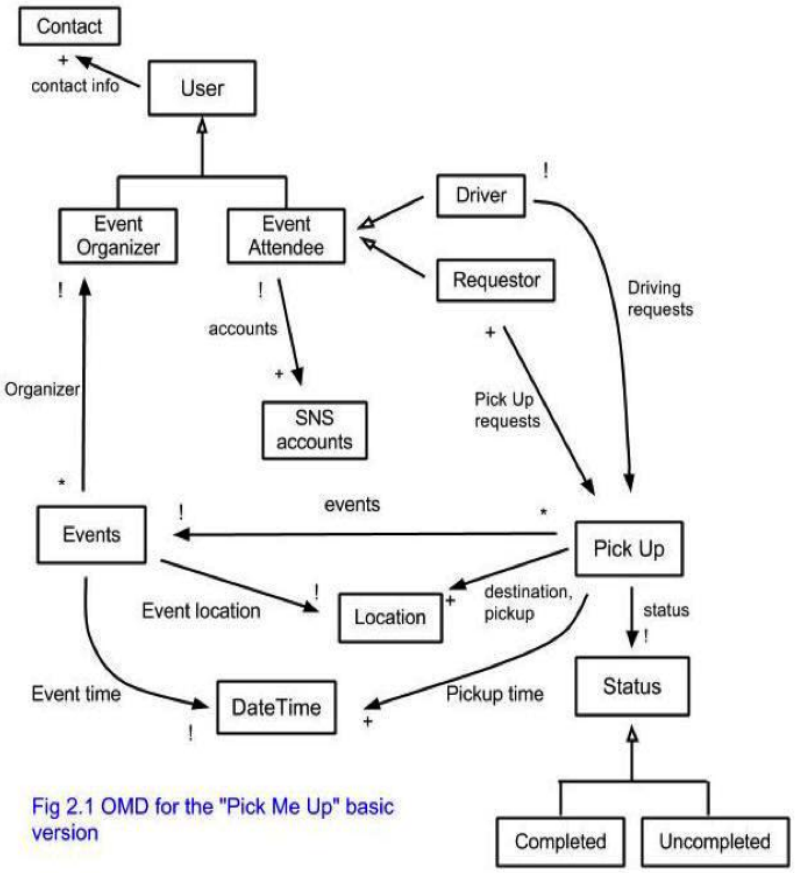

Fig 2.1 OMD for the "Pick Me Up" basic version

For the extended version, we are going to add several more details. The difference from the basic model would be **Verification Code** and **Payment account**.

- **Verification Code:** The verification code is intended for preventing bogus requests. The concept actually encompasses several verification tokens (as implied by the multiplicity in the OMD) to authenticate the requestors and the drivers who actually go to the event, and whether they have either picked up or been picked up according to the requests they posted on the site. Some of the verifications tokens include, ticket serial numbers and driver & requestor code.
- **Payment Account:** Users can have several payment accounts such as their online banking accounts, or paypal accounts so that all the monetary transactions can be carried out (cost-sharing, penalty, and etc..)

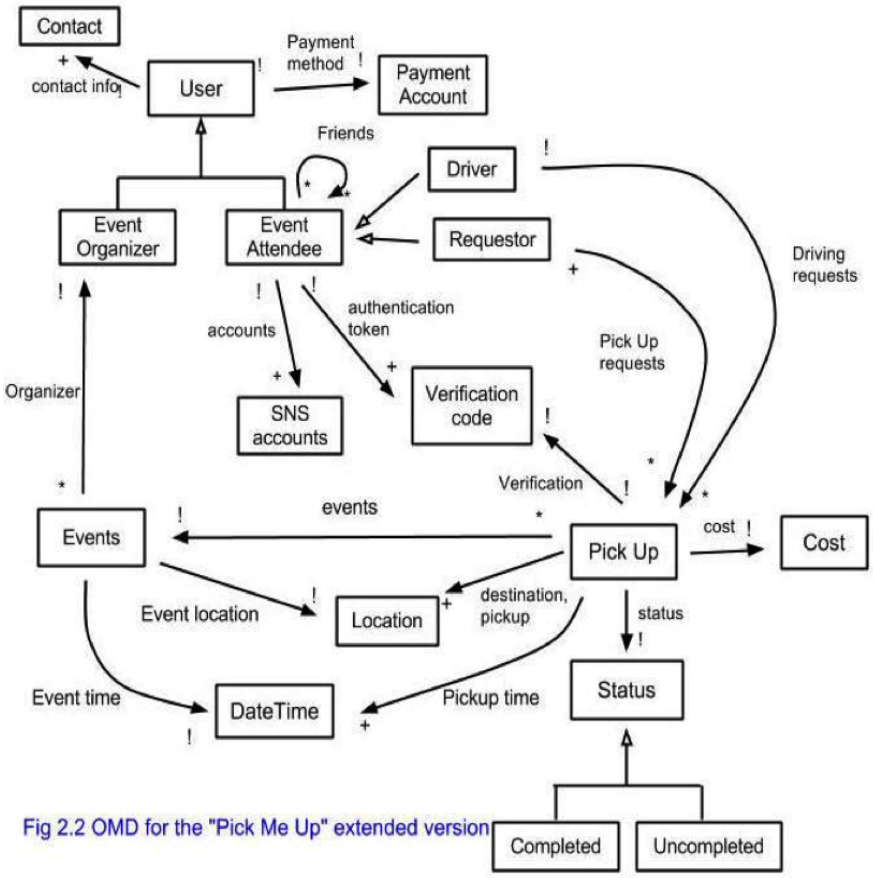

Fig 2.2 OMD for the "Pick Me Up" extended version

3 Behaviors

3.1 Functionality

The Event Organizer can:

- Add a new event
- Modify an existing event (name, date/time, location, genre)
- Delete an event
- Automatically detect location from string

The Event Attendee can:

- Add an event that the user is attending (requiring some verifications)
- Add a PickUp request, which includes data/time and location
- Modify PickUp request by adding name, location, details
- View suggestions based on request types (3 types: drive, want a drive, want to share cab)

- Accept or ignore suggestions and add participants to the suggestion lists

Extended features:
User can:

- Confirm and verify PickUp requests completion based on GPS data by running our application onSmartPhone.
- Add payment account
- View event statistics (number of registered attendants, number of car sharings)
- Modify presonal preferences (name, phone number, email address)

Event Organizer can integrate with existing event services (such as EventBrite, EventBee, TicketLeap)

3.2 Security

Pick-Me-Up! should satisfy the following key security requirements: (1) Users are protected (adversaries do not gain sensitive information, all participants are verified and trustworthy); (2) Users are trusted to follow through with their commitments.

3.2.1 Threat model: assumptions about attackers

- Attackers have physical access to clients but not servers (so expire).
- Attackers may snoop.
- Attackers know the underlying model designs.

3.2.2 Security risks (Application specific & Standard attacks)

- Adversary registers for event that they do not participate in. → Require participant for an event to verify their attendances (such as a confirmation code)
- Adversary registers fake events. → Only allow verified event organizer to register events.
- Users do not follow through with PickUp request. → Enforce no-show-up charges for participants on PickUp request.
- Adversary gains sensitive information (email address, phone number, credit card information) → Sensitive information are protected by authorization.
- XSS → sanitization; CSRF → form tokens; cookies → encrypt, expire; mass registration → captcha; SQL injection → data sanitization
- Security risks related to Facebook account → Facebook: authorization token with expiration, limited permission set.

3.2.3 To prevent bad access

- Authentication + access control on all sensitive actions (view, add, edit, delete).
- Expiring cookies and deleting on logout.

3.3 User Interface

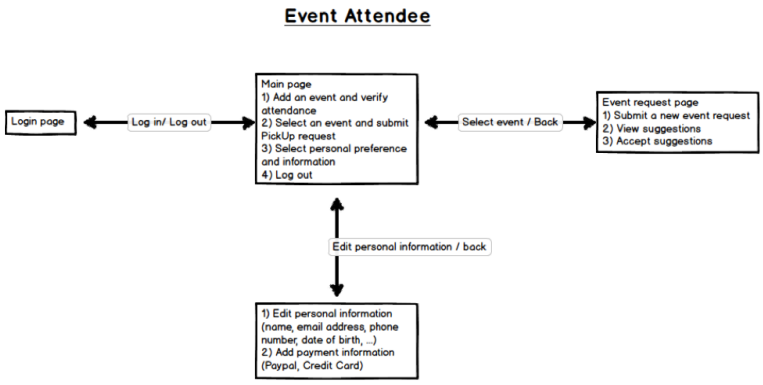

Figure 1: UI flow for event attendee

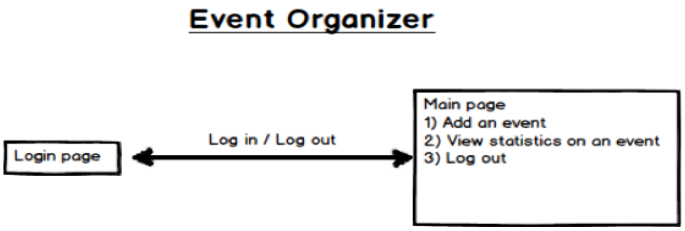

Figure 2: UI flow for event event organizer

Figure 3: UI flow for event attendee on smart phone

Selected UI mockups:

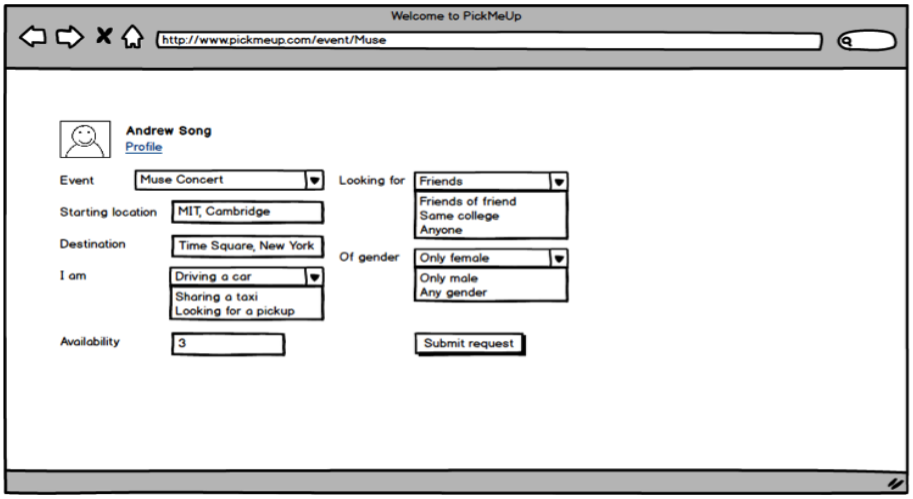

Figure 4: Event attendee's new PickUp request form

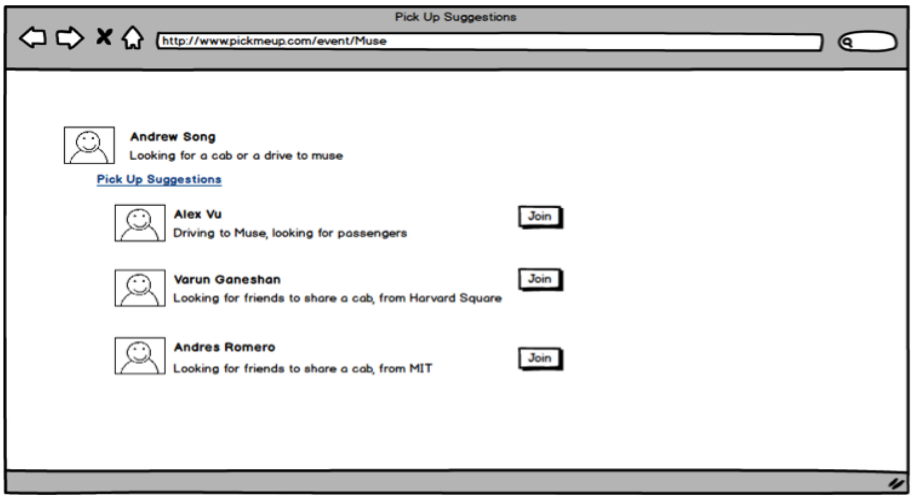

Figure 5: Event's attendant suggestion form

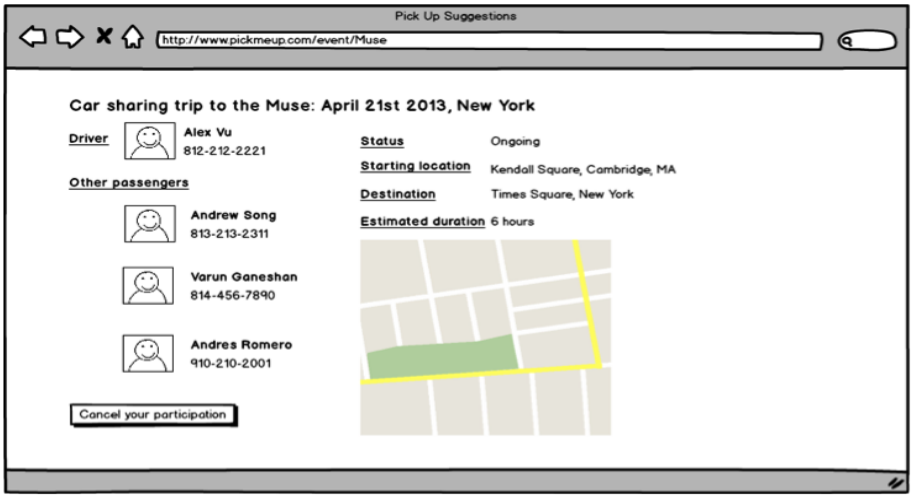

Figure 6: PickUp trip details

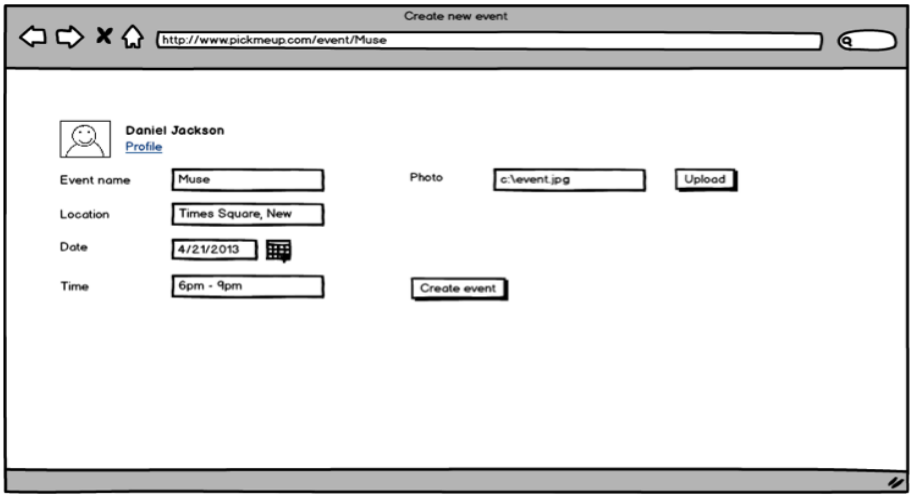

Figure 7: Creating a new event

4 Challenges

4.1 Types of log-in for organizers/attendees

- Both have custom authentication
- Both have third-party (Facebook) authentication
- Separate types of authentication (cusom for organizers and Facebook for attendees)

Solution: All these options are feasible; most event orgnaizers now have Facebook for their own events. However, For attendees, Facebook authenticatio is convenient and probably the favorite but they may be concerned about us retrieving too much of their personal information. We ask for minimal permissions for information possible. We also recognize that event organizers may require more custom functions and the number of organizers with a Facebook may still be limited. Hence, we choose the third option which we believe, best suits the preferences of our users.

4.2 Can users participate in multiple PickUps for a specific event?

- Yes - users can participate in multiple PickUps for an event
- No - for each event, users should be committed to only one pick up.

Solution: to make sure each user follows through with their commitment to an event, we require that each user is committed to one PickUp for each event.

4.3 Can PickUp request include multiple destinations (like a tour)?

- Yes. PickUp request can have multiple destinations (for example, following a musical tour)
- No. PickUp request can have a single destination.

Solution: Scheduling for multi-stop event is possible, but quite complicated. For simplicity, currently we only support single-destination event. A multi-stop event can be represented as a number of single-destination event.

4.4 How to deal with change in location (for cancellation of event, of course we can just cancel the pick up and notify users)?

- Still require users to pick up and drive to the new location
- Cancel the pick up and require users to post new pick up
- Change the location of the pick up and allow users the option commit or abandon within arbitrarily 2 days of change

Solution: The first and second options make it easy for developer. However, the first option is not reasonable because the initial commit is made on the point-in-time conditions. If one of the conditions changes, parties should have options to continue or terminate the contract. The second option will waste all our previous coordination effort. Hence we choose the third option.

4.5 Verify if the requester/driver turns up

- Set up a call center
- Post-GPS location through mobile device
- Not checking at all and let users figure out themselves

Solution: we need to implement this on mobile platform to make this feature truly effective. We will endeavor to use third option. We need to check this; the crux of the business is to ensure reliable service and commitment. Option 3 is out of the question. Option 1 is is simpler to implement on the developer' side. However, call center requires manual labor and checking (ask questions about the location). It is costly and not very reliable because it is possible for them to lie. Tracking GPS location is precise to a certain degree, definitely more reliable and objective. It is also cheap and more user friendly as users can simply press a button on their internet device to tell GPS location and generate automatic email to counter party to notify arrival rather than calling. We need to ensure that we have confirmation of message delivery. The risk lies in the implementations of mobile platform.

4.6 How do we give matching suggestions to users?

- Give suggestions to user based on social networking (such as Facebook friend list)
- Give suggestions to user based on geography
- Give suggestions to user based on other parameters (such as hobbies, interests and other preferences)

Solution: Given that these solutions are not mutually exclusive, when can perform multiple filters. For example, we can give suggestions to users that are based on both social networking (people who are friends with the user) and geography (who are in the current town). In our first iteration, we will implement the first option. For the extended version, we will also incorporate the second option.

4.7 Verify if the pick-up is complete

- Generate pickup code for requester and driver is required to enter this pick-up code.
- Call center
- Enough to confirm that both are at the location based on GPS tracking

Solution: Due to similar reasons above (verify if the driver/requestor turns up), we do not choose option 2. Option 3 is not reliable enough because they can both turn up but never meet each other. This gives us the chance to intervene and help. Overall, the procedure includes pressing a button to confirm GPS location, sending notification of arrival and entering pick-up code; this is simple and easy enough.

4.8 Mapping from physical entity to GPS coordinates and vice-versa

- Have an internal database of physical location and manually map the location to
- Use 3rd party services (such as Google Geocoding API and Reverse-Geocoding API)

Solution: Geocoding is the process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739). The Reverse-geocoding is the reverse process, converting coordinates into human-readable entities. Currently, Google allows us a query limit of 2,500 geolocation requests per day, which should be sufficient for this purpose. After some investigations, we can also use Geocoder gem with ease. The first option is not ideal in terms of scalability; although the internal database might be small and manageable in the early stages of the service, as the service gets bigger, the database will grow beyond our control.

5 Implementation plan

- April 14th to April 20th: Complete the design document and agree on the MVP
- April 21st to April 27th: First iteration of MVP
- April 28th to May 4th: Second iteration of MVP
- May 5th to May 11: Review and Critique

6 References

- Rails Coding Style: `http://pathfindersoftware.com/2008/10/elements-of-ruby-style/`
- Rails Idioms: `https://code.google.com/p/tokland/wiki/RubyIdioms`
- Geocoder Gem: `http://www.rubygeocoder.com/`