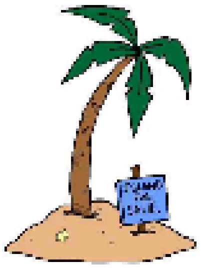
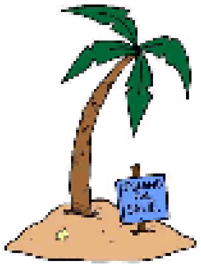
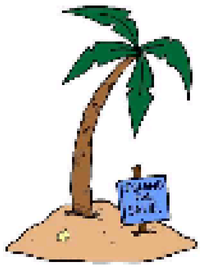
OpenTrip: An Open Protocol for the Interchange of Travel Information Among Rideshare Providers

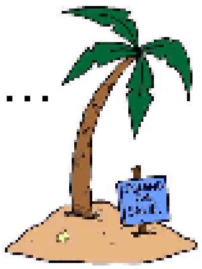
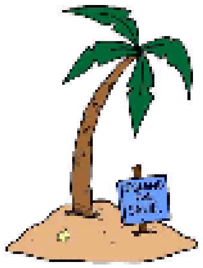
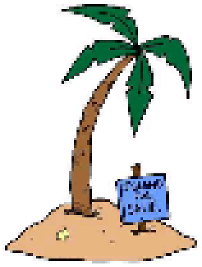
Carl Gorringe
carl@gorringe.org

http:// OpenTrip.info

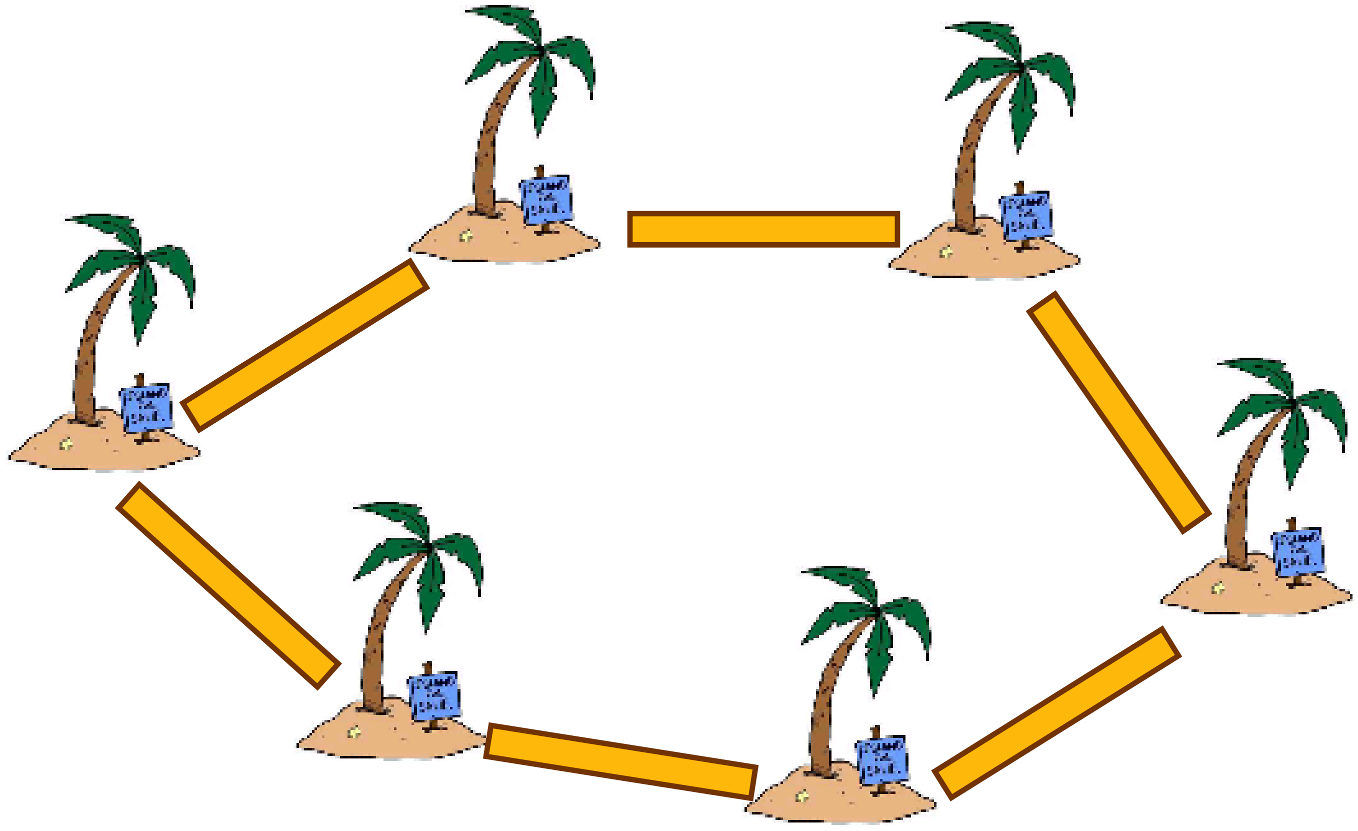
Rideshare services today

A multitude of rideshare providers,
each one on their own separate island…

Rideshare services tomorrow



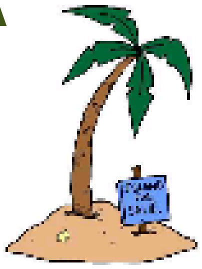
Let’s build some bridges!

Nice in theory but…

- Why should one service share its data with its competitor?
- Is there a good business case?
- Will this help solve, or at least get us closer to solving, the “critical mass” problem?

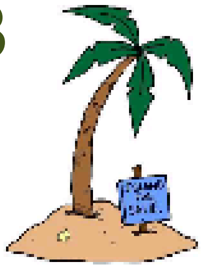
Let's see…

A



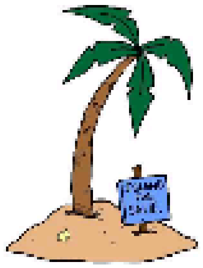
20,000 trips

B



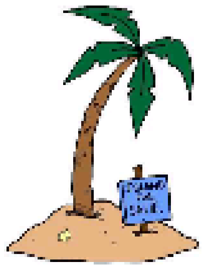
10,000 trips

C



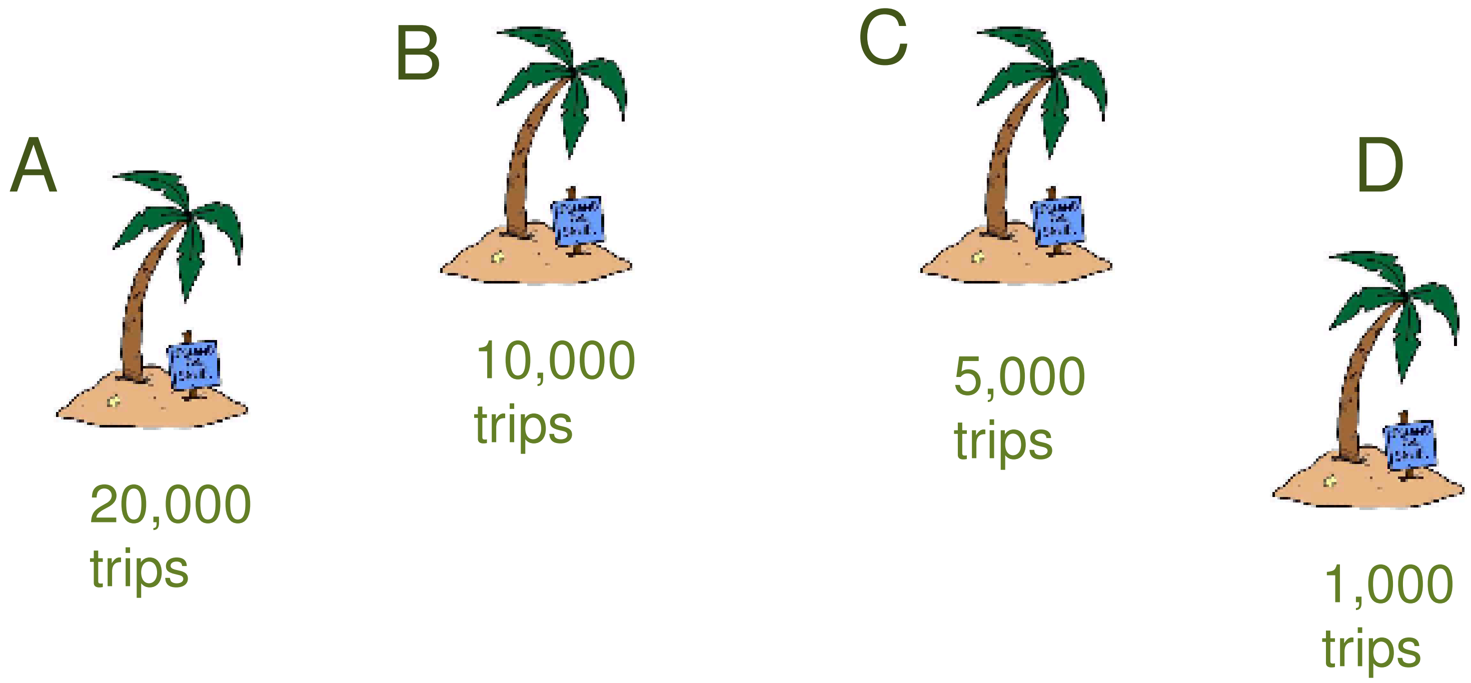
5,000 trips

D



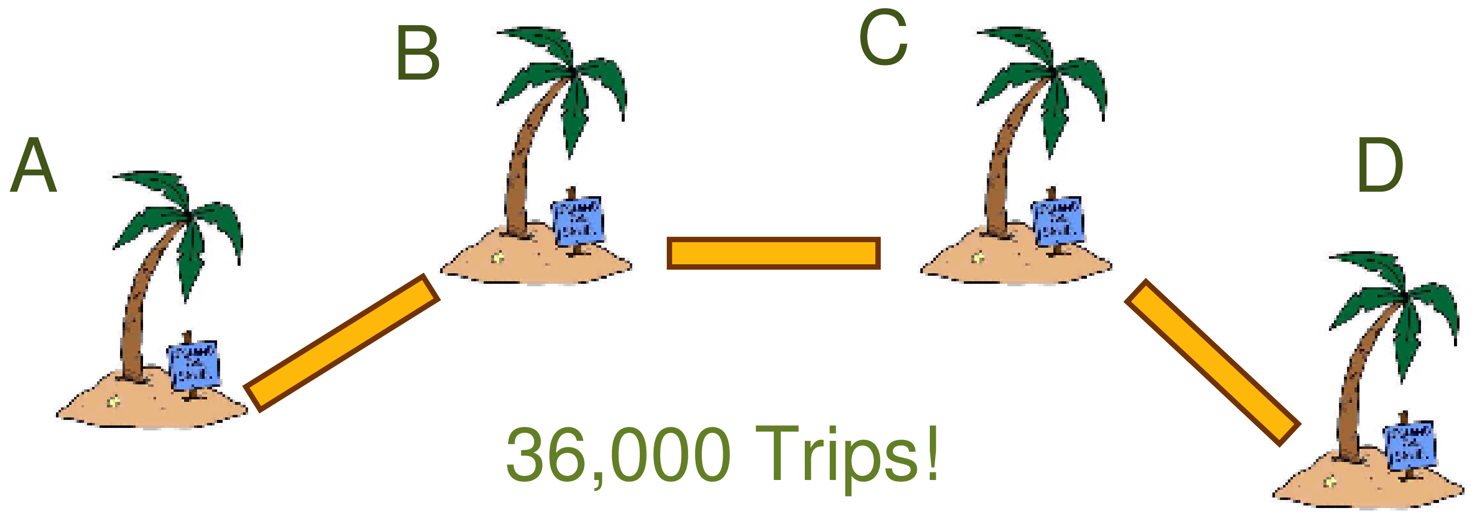
1,000 trips

Let's see…



A user who visited service C would have only 5,000 trips to search against. If the user didn’t find a match, he wouldn’t likely return. Thus the “Critical Mass” problem, where there aren’t enough trips within a given geographical area.

Let's see…



Now if every service shared a data feed, then any service that a user searched could provide a database of 36,000 trips, thereby increasing the odds of finding a match and the user returning to the website.

Potential Generators and Consumers of OpenTrip

- Social ridesharing services
- Car sharing
- Travel sites
- Taxi companies
- Air Taxi operators and brokers
- Travel planning kiosks
- Calendar software
- Event sites
- Any sort of trip planner
- Any sort of research project into travel patterns
- Location and Visitor Bureau sites
- Transit Agencies
- FAA, DOT, State and County Transportation Agencies

OpenTrip History

- First discussed at TransitCamp Bay Area in Palo Alto, California, Feb 23-24, 2008.
  (we called it "TripML")
- The Google Group *api-design-tripml* was created to continue the discussion.
- Further discussion at TransitCamp 2 in San Carlos, CA, Sept 13, 2008.
- More rideshare services were invited to the group in Dec '08 – Jan '09.

OpenTrip History (cont.)

- I developed an OpenTrip feed consumer and search engine for 511 Rideshare in first half of 2009.

- MIT Workshop on Dynamic Ridesharing in April ’09. Rideshare services expressed an interest in OpenTrip, but weren't entirely convinced. Perhaps they were waiting for 511 to make the first move.

OpenTrip History (cont.)

- 511 decided not to launch the project even though it was completed. They may yet decide to launch it at some point.
- Lack of interest from rideshare services meant that OpenTrip never really took off.

☹ Bummer ☹

OpenTrip History (cont.)

- I gave up on the project, however…
- In 2010, Daniel Graziotin, a grad student in Italy developed Dycapo, a dynamic carpool RESTful JSON protocol, based off of OpenTrip.

☺ Yea for Open Source! ☺

Basic Principles

- Open Standards -- *anyone may use freely.*
- Published Openly.
- Distributed Architecture.
    - *Avoid centralized databases and points of failure!*
- Extensible for future needs.

Layers of OpenTrip

- OpenTrip Core
  - Defines the data structures for our trip data.
  - Simple mechanism of publishing as an Atom/RSS feed, useful for traditional rideshare websites.
  - Doesn't require authentication. (keeps it simple)
- OpenTrip Dynamic API
  - Suited for dynamic ridesharing.
  - Real time updates, resource allocation, authentication and messaging.

Layers of OpenTrip (cont.)

- OpenTrip Ping
  - Simple mechanism to inform a feed consumer that there is an update to the feed.
- OpenTrip Search
  - Uses OpenSearch to return a feed of ride matches from participating websites.
- Resource Discovery
  - Find out what services are offered on individual servers, and a registry of all servers.

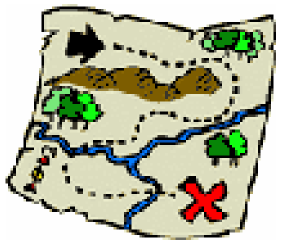
What Data?

There are 2 basic pieces of data required for a ride match:

What Data?

There are 2 basic pieces of data required for a ride match:

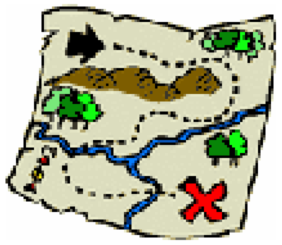
1.



Location & Time of Trip

What Data?

There are 2 basic pieces of data required for a ride match:

1.  Location & Time of Trip

2. Means of Contacting Users

What Data?

- Both pieces are required or else a ride match won’t work.

What Data?

- Both pieces are required or else a ride match won’t work.
- A ridesharing service could share just one of the two pieces of data: The location and time of the trip.

What Data?

- Both pieces are required or else a ride match won’t work.
- A ridesharing service could share just one of the two pieces of data: The location and time of the trip.
- A competing service could then match riders with drivers, but would have to direct the user to the originating service to obtain the contact information.

OpenTrip Core:
Minimum requirements in a data feed

1. Origin & Destination Locations, preferably as Lat / Lon coordinates.
2. Date & Time of ride, one-time or recurring.
3. User Preference: Drive, Ride, or Both.
4. A unique *Trip ID.*
5. Expiration date.

OpenTrip Core: Minimum requirements in a data feed (cont.)

1. One of either of these:
   1. For websites, a URL pointing to detailed information on source website, which would include a means to contact carpooler.

OpenTrip Core: Minimum requirements in a data feed (cont.)

1. One of either of these:
   1. For websites, a URL pointing to detailed information on source website, which would include a means to contact carpooler.
   2. For dynamic rides, a *Contact ID* or *User ID*, which may be used to contact the carpooler in real-time through the use of an API on the source's server.

OpenTrip Core in Detail: (1 of 6)

Skeleton of an OpenTrip Atom Feed

```
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:g="http://www.georss.org/georss"
      xmlns:t="http://opentrip.info/-/opentrip/0.1/">
  <title>Example Feed</title>
  <link rel="self“
    href="http://example.com/feeds/foobar.xml"/>
  <id>urn:guid:example.com:ABCDEFG</id>
  <updated>2009-01-01T01:23:45Z</updated>
  <author><name></name></author>
  <entry> ... </entry>
  <entry> ... </entry>
...
</feed>
```

OpenTrip Core in Detail: (2 of 6)

An Entry is a single, round-trip or recurring Trip.

```
<entry>
  <title>I need a ride</title>
  <link href="http://example.com/postings/123456789.html"/>
  <id>urn:guid:example.com:123456789</id>
  <published>2009-01-01T01:23:45Z</published>
  <updated>2009-01-01T01:23:45Z</updated>
  <t:expires>2009-09-01T01:23:45Z</t:expires>
  <content>I'm looking for a ride to work!</content>
  <t:location> ... </t:location>
  <author> ... </author>
  <t:prefs> ... </t:prefs>
  <t:mode> ... </t:mode>
</entry>
```

- **Published:** Date when trip created.
- **Updated:** Date when trip last modified.
- **Expires:** Date used to prevent stale entries.

OpenTrip Core in Detail: (3 of 6)

Location Construct

```
<t:location label="Home">
  <t:town>Oakland</t:town>
  <t:region>CA</t:region>
  <t:country>US</t:country>
  <g:point>37.774311 -122.214746</g:point>
  <t:leaves recurs="weekly" days="MTWHF" offset="30">
    2009-04-01T08:00:00</t:leaves>
  <t:returns recurs="weekly" days="MTWHF" offset="30">
    2009-04-01T18:00:00</t:returns>
</t:location>
```

Includes Lat / Lon coordinates, an address,
and one-time, round-trip, or recurring date-times.

A location in 4-D space!

OpenTrip Core in Detail: (4 of 6)

Person Construct

```
<author>
  <name>John Doe</name>
  <email>john@example.com</email>
  <uri>http://example.com/profile123.html</uri>
  <t:uri>http://example.org/alternate123.html</t:uri>
  <t:phone label="mobile">510-555-1234</t:phone>
  <t:age>25</t:age>
  <t:gender>male</t:gender>
</author>
```

Name, email, phone,

website *(e.g. Facebook profile),*

age, gender, etc...

OpenTrip Core in Detail: (5 of 6)

Preference Construct

```
<t:prefs>
  <t:drive/><t:ride/>
  <t:age>18-30</t:age>
  <t:gender>female</t:gender>
  <t:nonsmoking/>
</t:prefs>
```

Personal preferences of the rider or driver,
for filtering of ride matches.

OpenTrip Core in Detail: (6 of 6)

Mode Construct

```
<t:mode kind="auto">
  <t:cost kind="USD">2.00</t:cost>
  <t:capacity>2</t:capacity>
  <t:vacancy>1</t:vacancy>
  <t:make>Tesla</t:make>
  <t:model>Roadster</t:model>
  <t:year>2009</t:year>
  <t:color>Red</t:color>
  <t:lic>ABCD123</t:lic>
</t:mode>
```

Describes the automobile being used,
or other mode of transportation.

Example Modes:

auto, taxi, van, bus, rail, ferry, walk, airplane,
slug (slug-line or casual carpool pickup spot)

What has been done so far…

- OpenTrip Core Draft spec posted at: http://www.opentrip.info
- 511 Rideshare feed consumer and search engine. Could they release an OpenTrip feed in the future?
- Dycapo API spec, documentation, and open-source code at:

  http://dycapo.org

Next Steps

- Should we pursue this any further?
- Pressure rideshare services to open up their data?
- Continue working on the Dynamic Ridesharing API and/or develop open-source software?
- Find sources of funding?

Thank You

Carl Gorringe
carl@gorringe.org

http:// OpenTrip.info