Strava

Overview

This is a case study of Strava, a personal-fitness tracking app that lets you record, analyze, and share your running and cycling data. The resources organized by Strava are the metadata associated with a workout; the social interactions users have on their site; and the map routes created by users. Strava does large-scale digital organizing and is just one of hundreds of apps within the fitness tracking domain.

What distinguishes Strava from its competitors is its elegant user interface and "social fitness" platform. In addition to providing useful statistics and visualizations of exercise data, Strava allows users to socialize, motivate, and compete via online and mobile apps. Their business model is largely based on selling premium features in their mobile and web-based apps. Individuals can use these enhanced features to improve their performance and gain new insights into their workout routines. In addition to the mobile app and web client, Strava provides an API that allows 3rd party apps to integrate with their services. These services allow users to share their data on personal sites, provide enhanced analytical tools, and many other features.

Recently, the city of Portland, OR purchased an annual subscription to Strava's data through a new program called Strava Metro. Armed with anonymised data about city ridership, officials are now able to plan the construction of new bike lanes with greater confidence in their decisions. With dozens of other cities also interested in Strava Metro, I became interested in the use of personal fitness data for civic planning. Since Strava's API wasn't initially designed with civic planners in mind, I'm interested in exploring how services like Strava can transform their APIs to allow civic planners to make better use of their data.

Tags: Computational and Automated Resource Description (chapter 4), *Naming Resources (chapter 3.4)*

What is being organized?

Strava designed three user interfaces – web app, API, mobile app – to meet the needs of its users. In addition to the exercise metadata, social interactions, and labeled map routes described above, Strava also organizes nine resources through their API: an authentication protocol (OAuth2), athletes (Strava users); activities (runs, rides, etc.); clubs (groups of athletes); gear (equipment used during activity); segments (specific sections of road); segment efforts (portion of a ride in a segment); streams (raw data associated with an activity); and uploads (submitting raw person data from a 3rd party to Strava). Each API resource includes several resource descriptions. For example, the "athlete" resource includes descriptions such as name, sex, location, and the number of friends or followers. Each athlete can also be associated with a clubs, bikes, and even shoes. The "activity" resource includes descriptions of a specific activity recorded by the user, and is associated with the "segment effort" and "athlete" resources.

The mobile app includes only a subset of the data from the full-scale web version, but provides real-time exercise data that is organized and communicated back to the user visually. The web client provides the greatest visual experience for the user while the API provides highly granular data that can be used as an input for a separate computational task.

Why is it being organized?
Exercise data is being organized to help users gain insights into their exercise routines and promote interaction with the app. The API resources are being organized to facilitate the exchange of data between software applications, which are primarily focused on enhancing the individual user's experience. Most Strava users aren't directly aware of the existence of an API, and may only experience it when asked by 3rd party software for permission to access their data. The Strava API is of interest primarily to software developers who are building applications. By providing a well-documented and structured API, Strava can empower software developers to building supportive interactions with existing user data.

How much is it being organized?
Users must start the app in order for Strava to begin collecting exercise data, which is why Strava must design an experience that promotes interactions with their service. Strava also allows users to bulk upload data collected for other decides or apps, and also to delete specific resources -- e.g. comments with specific users, workouts, and mapped routes. Since the data is collected through a mobile and web apps, Strava can quickly organize, analyze, and display the data back to the user. One of Strava's greatest competitive advantages is their ability to organize all the data collected and generate new resource descriptions effortlessly. With a large and highly engaged user community, Strava is able to get hand labeled route and segment data for free.

When is it being organized?
Resources are organized the moment the app is started. Geo-location data is fed to Strava through the user's smart phone or GPS device. By combining resource descriptions like start and end time, Strava can create new descriptions like average speed. While riding a bike, users can see their location, elapsed time, average speed, acceleration, elevation, and even heart rate directly on their mobile device. By organizing and displaying this information in real-time, users can adjust their exercise accordingly. Once the ride is over, data is sent to Strava's servers and processed. A detailed report about their routine can be found on their mobile device or web app, with raw data made available through the API.

How or by whom is it being organized?
Resources are organized both by Strava (computationally) and by users. Following the ride or run, users must enter a name to identify their workout. Users can also hand labels common routes or stretches of road with unique identifiers. For example, the stretch of Telegraph between Oregon St. and Haste is known as the

"Sather Road Climb." There is no controlled vocabulary for assigning segment identifiers by users. Strava organizes the data visually on a dashboard for users to interact with and digitally through the API for developers to use.

Other considerations.

Strava is not making use of any air quality or traffic data. Integrating these sources of information could provide additional value to users who are already accessing it through a separate service. Strava could also categorize and rank them by level of difficulty. These additional descriptions would allow users yet another means of searching and retrieving.

Strava API Transformation

Strava recently created a new service called Strava Metro for "departments of transportation and city planners, as well as advocacy groups and corporations [to] make informed and effective decisions when planning, maintaining, and upgrading cycling and pedestrian corridors."[1] Anonymized cycling data is provided to cities in bulk and is formatted using GIS software. With some minor modifications to the Strava API, cities can gain direct access to anonymized data in real-time.

The "Activities" API resource contains 50 resource descriptions.[2] Biometric descriptions (e.g. "calories" and "heart rate") and social descriptions (e.g. "has_kudoed", "comment_count", "photo_count") are unnecessary. Descriptions for "average_speed", longitude and latitude ("start_latlng", "end_latlng"), and "start_date" are important, however. In the Appendix, I highlight in yellow important descriptions and in red, unimportant descriptions.

Since the data provided through the API is very granular, the resolution could be reduced to protect the anonymity of users. The "athlete" object provides a unique identifier for each user. Instead, Strava could populate this field with a random string or remove it entirely. City planners could use the ride ID instead to track the progress of the user.

City planners may be interested primarily in how commuters use roads to travel between home and work. Strava allows users to manually label the type of ride, which is also provided through the API. In addition to this labeled data, Strava could use clustering algorithms to classify the ride as a commute if users aren't completing this field manually. Labeled data for the "commute" description could be used to predict whether an unlabeled ride is also a type of commute. For planners who need greater certainty in the type of ride, Strava could easily provide the prediction value for their binary classification algorithm.

By examining riding patterns through city streets, Strava could also detect unsafe intersections. By creating a new description for street interaction and labeling it with the name of the streets, time, and a global safety score, Strava could provide additional value to city planners. City planners could then cross reference these intersections with both longitude/latitude and labeled street name with internal datasets for road safety.

With just a few minor adjsutments, Strava's powerful API could give cities greater access to information while still preserving the privacy of its users.

---

[1] http://metro.strava.com/

[2] http://strava.github.io/api/v3/activities/

Strava API Transformation

Appendix

id:	**integer**
resource_state:	**integer** indicates level of detail
external_id:	**string** provided at upload
athlete:	**object** meta or summary representation of the athlete
name:	**string**
description:	**string**
distance:	**float** meters
moving_time:	**integer** seconds
elapsed_time:	**integer** seconds
total_elevation_gain:	**float** meters
type:	**string** activity type, ie. ride, run, swim, etc.
start_date:	**time string**
start_date_local:	**time string**
time_zone:	**string**
start_latlng:	**[latitude, longitude]**
end_latlng:	**[latitude, longitude]**
location_city:	**string**
location_state:	**string**
location_country:	**string**
achievement_count:	**integer**
kudos_count:	**integer**
comment_count:	**integer**
athlete_count:	**integer**
photo_count:	**integer**
map:	**object** detailed representation of the route
trainer:	**boolean**
commute:	**boolean**
manual:	**boolean**
private:	**boolean**
flagged:	**boolean**
workout_type:	**integer** for runs only, 0 -> 'default', 1 -> 'race', 2 -> 'long run', 3 ->

Strava API Transformation

	'intervals'
gear_id:	**string** corresponds to a bike or pair of shoes included in athlete details
gear:	**object** gear summary
average_speed:	**float** meters per second
max_speed:	**float** meters per second
average_cadence:	**float** RPM, if provided at upload
average_temp:	**integer** degrees Celsius, if provided at upload
average_watts:	**float** rides only
weighted_average_watts:	**integer** rides with power meter data only similar to xPower or Normalized Power
kilojoules:	**float** rides only uses estimated power if necessary
device_watts:	**boolean** true if the watts are from a power meter, false if estimated
average_heartrate:	**float** only if recorded with heartrate average over moving portion
max_heartrate:	**integer** only if recorded with heartrate
calories:	**float** kilocalories, uses kilojoules for rides and speed/pace for runs
truncated:	**integer** only present if activity is owned by authenticated athlete, returns 0 if not truncated by privacy zones
has_kudoed:	**boolean** if the authenticated athlete has kudoed this activity
segment_efforts:	**array of objects** array of summary representations of the segment efforts, segment effort ids must be represented as 64-bit datatypes
splits_metric:	**array of metric split summaries** running activities only
splits_standard:	**array of standard split summaries** running activities only
best_efforts:	**array of best effort summaries** running activities only