Prevención de lugares problemáticos en ruteos de vehículos

Santiago Castro, Jennifer Esteche, Bernardo Larrosa y Romina Romero
Grupo 05

***Resumen*—En el presente documento se aborda el diseño de un sistema que resuelve el problema de hallar rutas que eviten ciertos puntos relacionados a lugares problemáticos de tránsito. A su vez, el sistema recupera esta información de problemas de tránsito desde la cuenta de Twitter *@chanchosUY* y luego es clasificada, indexada y almacenada en una base de datos no relacional. Se brinda también una aplicación móvil para que los usuarios puedan recorrer una ruta entre dos puntos, indicados por ellos, evitando los puntos problemáticos registrados.**

***Palabras clave*—tránsito, inspector, ruteo, accidente.**

I. Introducción

ES un problema común que cuando los conductores circulan por las ciudades se encuentren con problemas de tráfico generados por accidentes, trancazos, semáforos rotos y demás. Muchas veces se recurre al noticiero o inclusive a la prensa escrita para informarse al respecto, pero de esta manera no pueden obtenerse datos recientes, al instante.

En setiembre del 2011 se creó la cuenta de *Twitter* *@chanchosUY* [3] con el propósito de informar inicialmente la ubicación de los inspectores de tránsito en nuestro país de forma colaborativa, y con el tiempo también se agregaron notificaciones de distintos tipos de problemas de tránsito adicionales. Esto tiene la gran ventaja de que cualquiera puede enviar un mensaje avisando que hay lugares con problemas de tránsito. Para acceder a dicha información se utiliza generalmente un teléfono móvil con la aplicación de Twitter, o con alguna otra aplicación para dicho cometido, permitiendo que sea consultada en cualquier momento y accedida casi al instante en el que ocurrió el problema. Las personas reportan dificultades relevantes, como accidentes de tránsito, calles cortadas o semáforos rotos, pero dicha información está dispersa en todos los mensajes enviados a *@chanchosUY*, sin ninguna categorización que facilite la búsqueda por zona o por recorrido a realizar, ni siquiera por tipo de problema. Si se desea ir desde un punto a otro de la ciudad, hay que revisar todos los mensajes para informarse sobre alguna eventualidad que pueda existir en el camino, lo que resulta impráctico y engorroso. En otras palabras, es difícil saber entre tanta información cuáles de los problemas afectan a una persona particular en su camino hacia el lugar que desea llegar.

Se propone realizar un sistema que resuelva el problema de hallar rutas que eviten ciertos puntos relacionados a lugares problemáticos. El proyecto consiste en un sistema que recupera información de problemas de tránsito y brinda una aplicación móvil para que los usuarios puedan recorrer una ruta entre dos puntos, indicados por ellos, evitando los puntos problemáticos registrados. La información se extrae desde la cuenta de Twitter mencionada y luego es clasificada, indexada y almacenada en una base de datos no relacional.

En resumen se quieren atacar dos problemas bien definidos: el primero es relacionar los tweets de la cuenta de *@chanchosUY* con tipos de problemas y con puntos geográficos (establecer latitud y longitud) y el segundo es armar una ruta que esquive los mismos, dados un punto de origen y otro de destino.

Adicionalmente, cabe destacar que los propios fundadores de *@chanchosUY* crearon una aplicación móvil llamada *Oincs* [10] para ubicar en un mapa los problemas (y abarcar el primer problema mencionado), pero ésta no utiliza los datos de la cuenta de Twitter, sino que insta a los usuarios a que ingresen la información desglosada, desperdiciando de alguna forma los *tweets* existentes en la cuenta.

II. Solución planteada

II-A. Arquitectura del sistema

En la Figura 1 se muestra el diseño a gran escala del sistema construido.

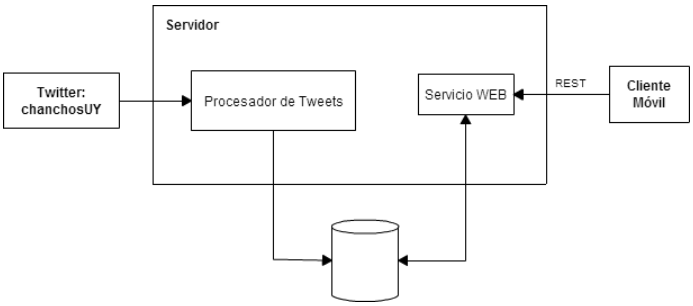

Figura 1. Arquitectura del sistema construido

II-B. Servidor

Se utiliza una computadora dedicada a recibir e indexar los nuevos tweets de la cuenta *@chanchosUY* (y desde una cuenta auxiliar utilizada para pruebas) para permitir obtener, mediante servicios REST, todos los problemas ingresados las últimas 24 horas con su ubicación geográfica y una ruta alternativa dados los puntos de origen y destino, teniendo en cuenta dichos problemas.

Al iniciar el servidor, se obtienen de las cuentas de twitter los últimos tweets ingresados que no se encuentran ya en la base, los cuales son indexados y salvados. Luego, se establece

permanentemente una conexión con twitter, por medio de la cual se irán obteniendo (e indexando) los tweets cada vez que son publicados por los usuarios.

El modelo propuesto en la solución se presenta en la Figura 2.

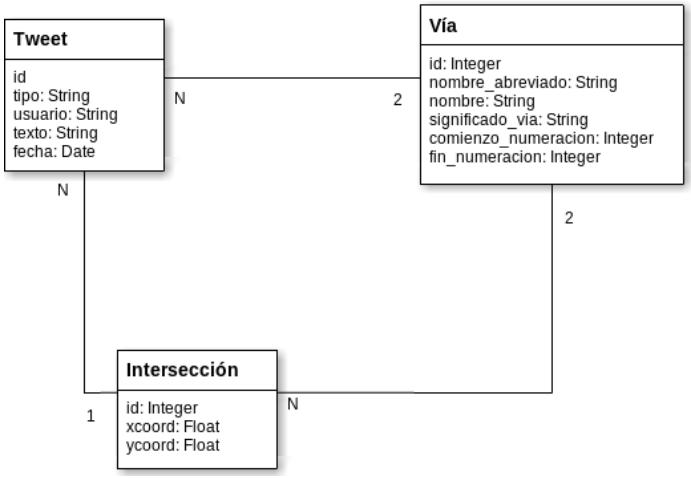

Figura 2. Modelo utilizado para la solución de la indexación

Los tweets son interpretados para poder saber en qué lugar ocurre el problema y de qué tipo es (inspectores de tránsito, accidentes, calles cortadas, manifestaciones). Para ello, en esta versión de prueba de concepto, se utiliza la base de datos de vías brindada por la Intendencia de Montevideo [7], que cuenta con todas las vías de la ciudad así como sus abreviaciones, entre otros datos. Luego, se buscan en cada tweet, mediante expresiones regulares, todas las vías que aparecen escritas, tal cual lo indica la base de la intendencia, y se las relaciona en la base con las mismas.

A su vez, se utiliza la información de los cruces de las vías provista por la Intendencia de Montevideo [5]. Dicha información contiene los nombres de las vías de cada intersección y las coordenadas en UTM (Universal Transverse Mercator Coordinate System) del cruce (se implementó la conversión de las mismas a coordenadas geográficas). En una primera instancia, se planeó utilizar la API de Google [4] para obtener las coordenadas (latitud y longitud) del cruce, pero debido a que Google no funciona correctamente con los cruces de las calles de Uruguay, se optó por esta solución. Cada intersección está asociada a dos vías y cada tweet se asocia a una intersección.

Para obtener el tipo de problema se utilizaron expresiones regulares, buscando de esta forma ciertas palabras claves en el texto del tweet, de no encontrar ninguna se le asignó la categoría "otro".

Cada vez que se recibe un tweet, el mismo es indexado, asociándole la intersección correspondiente (según las vías a las que haga referencia) y el tipo. En caso de no encontrar una intersección, es decir, no referencia en el texto a dos vías, no se le asocia ninguna y por lo tanto el tweet es descartado como problema debido a que no es posible ubicarlo geográficamente.

Para ofrecer el servicio de obtener todos los problemas de las últimas 24 horas, se retorna una lista de las coordenadas geográficas, el tipo y el texto de todos los tweets cuya fecha de creación es menor a un día. De esta forma, desde el móvil, pueden mostrarse en el mapa los problemas con sus respectivos tipos. Por otro lado, para poder ofrecer una ruta alternativa, en primer lugar se obtiene la ruta óptima brindada por Google [4] entre el origen y el destino. Luego, se hallan todas las intersecciones (cruces de calles) pertenecientes a la misma que presentan problemas para así poder evitarlas. Como Google no presenta una opción para obtener una ruta que evite ciertos puntos, pero sí una opción para que pase por otros (denominados "waypoints"), se obtiene una nueva ruta cuyos waypoints son puntos cercanos a los problemas. Esto se realiza de forma iterativa hasta que la nueva ruta no contenga incidentes.

Dicha ruta cuenta con "steps" que representan los puntos en donde hay un cambio de dirección. Por este motivo, las rutas que brinda Google no tienen marcadas todas las intersecciones por las que pasa. Para hallar las mismas, se toma cada segmento entre un step y el siguiente y se hallan las intersecciones que se encuentran en una región centrada en dicho segmento, como muestra la Figura 3.

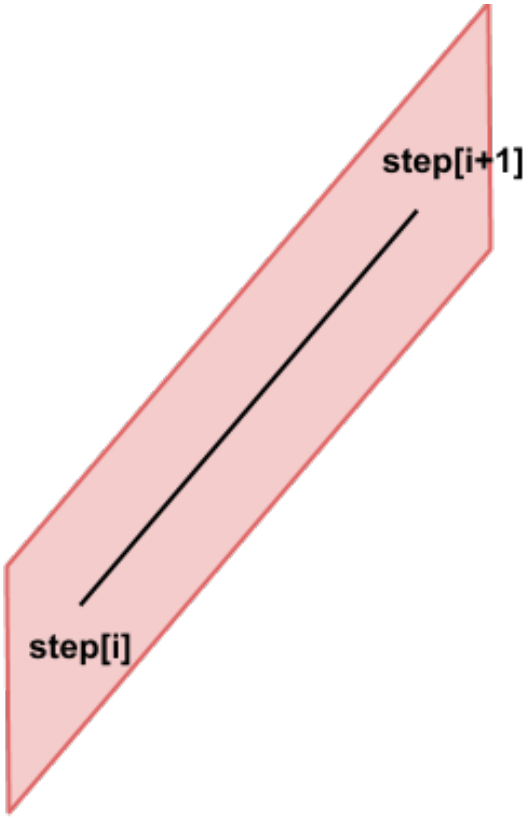

Figura 3.

Luego de hallar las intersecciones de la ruta, se buscan aquellas que contienen problemas, o sea, las que tienen asociacio algún tweet con fecha de creación menor a 24 horas. Estas últimas, son las que se trata de evitar seteando como waypoints sus coordenadas más un cierto delta. De esta forma, se continúa hasta que no haya incidentes en la ruta, devolviendo así un recorrido alternativo.

II-C. *Cliente móvil*

Consiste en una aplicación *Android* [1] que consume los servicios REST. Dicha aplicación le ofrece al usuario un camino para llegar a donde desea desde el punto de origen indicado, evitando pasar por los lugares problemáticos. En la Figura 4 se muestra una captura de pantalla de la misma.

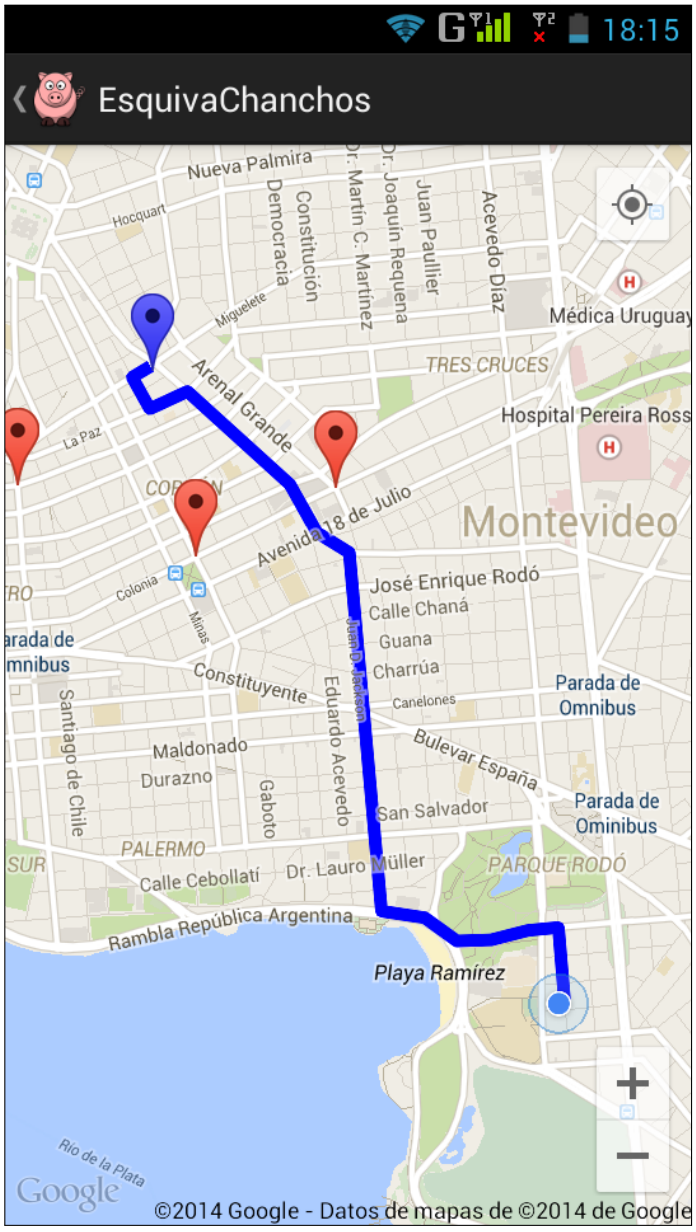

Figura 4. Captura de pantalla de la aplicación móvil sugiriendo una ruta que evita ciertos puntos problemáticos.

Para ello, se utiliza la API de Google Maps de forma de obtener la mejor ruta, y luego se le pide al servidor que modifique la ruta para poder evitar los tipos de lugares mencionados.

El usuario elige en el mapa el punto a donde desea ir manteniendo apretada la pantalla en dicho lugar para seleccionar y luego apretando nuevamente para calcular la ruta. El punto de origen es el de la persona, que se obtiene a través del GPS del celular.

En la Figura 5 se observan las opciones disponibles de tipos de problemas a evitar. El usuario puede elegir uno o más entre inspector, accidente, corte, trancazo u otros.

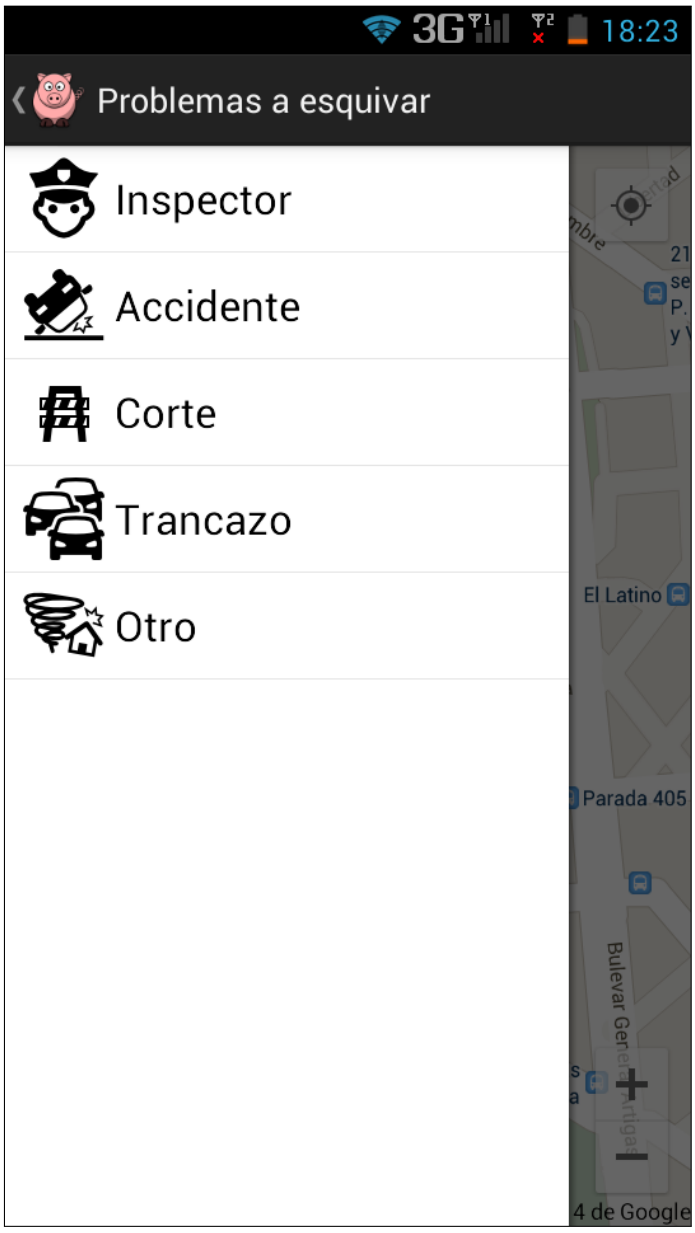

Figura 5. Captura de pantalla de la aplicación móvil que muestra distintos tipos de problemas que pueden ser filtrados.

III. Herramientas utilizadas

Se utiliza el *Lenguaje de Programación Ruby* [8] para traer los tweets de la cuenta, utilizando la *API Streaming* de Twitter [14] mediante la biblioteca *tweetstream* [13]. El sistema gestor de base de datos utilizado es *MongoDB* [9]. Para ofrecer el servicio web se hace mediante REST y se usa el framework *Sinatra* [12] para Ruby.

La herramienta utilizada para desarrollar en Android es *Android Studio* [2]. Se utiliza la librería de Google [4] Maps para mostrar en el mapa los puntos. Como cliente REST se utiliza *retrofit* [11].

IV. Resultados

Los resultados obtenidos son considerados buenos, logrando indexar una gran cantidad de tweets de la cuenta de Twitter *@chanchosUY*, a pesar de que no se logró indexar todos debido a que en el texto puede no aparecer dos vías tal cual están guardadas en la base. De todas formas, se consiguió ubicar geográficamente la totalidad de los tweets que fueron indexados, mostrándolos en el mapa de la aplicación correctamente.

Por otro lado, la aplicación da como resultado rutas que parecen ser cercanas a las óptimas, en cuanto a distancia, más allá que se le establezcan restricciones. Esto es bueno ya que parecería ser que no se pierden soluciones de calidad a pesar de que se tengan que descartar algunas soluciones buenas.

V. Conclusiones

No es una tarea fácil interpretar los tweets de las personas para analizarlos semánticamente de forma de encontrar calles o tipos de problemas. Por ejemplo, la gente comete errores ortográficos al escribir un tweet para indicar un problema o llama a las cosas por otros nombres, lo cual dificulta la tarea. A su vez, no siempre dicen las calles en donde se encuentra el problema, sino que a veces mencionan lugares específicos o referencias locales, como monumentos o plazas. Inclusive a veces eligen enviar imágenes para describir el evento, sin mucha descripción, como por ejemplo una foto de inspectores en algún lugar de la ciudad, y esto hace aún más compleja la tarea. También es posible que palabras comunes sean interpretadas como calles por tener nombres parecidos o que sea difícil evitar ciertos puntos para llegar a un lugar, porque pueden no quedar más caminos. Más allá de esto, la aplicación parece ubicar muy bien los puntos que son referenciados.

Sin embargo, la aplicación no tiene muy buen acierto al intentar clasificar la categoría del problema. Esto es debido a que las personas tienen una gran cantidad de formas distintas y creativas de hacer referencia a los problemas, en particular a los inspectores de tránsito (se les llama “chanchos”, “chanchitos”, “oink oink”, “chiquero”, etc), y no son interpretadas correctamente por la aplicación.

VI. Trabajo futuro

Respecto a la parte de indexación, sería de interés poder mejorar el análisis de los tweets. Por ejemplo, se podría reconocer lugares específicos, como plazas, shoppings, o incluso referencias más informales pero que todos entendemos, como “estadio” o “las canteras”. En particular, la Intendencia de Montevideo [6] brinda información sobre puntos de interés como “Estadio Centenario” o “Montevideo Shopping”, por lo que se podría utilizar dicha información.

Sería interesante poder elegir otro punto de origen en la aplicación que no sea el detectado por el GPS. Los usuarios podrían querer planificar rutas que no son para el momento actual, o que son para otras personas tal vez.

Adicionalmente sería de interés poder buscar direcciones escribiendo la intersección con el teclado del celular, o más aún con la dirección exacta o con el nombre del lugar al que se quiere ir.

Referencias

[1] *Android*. *Android*. URL: https://www.android.com/ (accedido en diciembre de 2014).

[2] *Android Studio*. *Android Studio | Android Developers*. URL: https://developer.android.com/sdk/installing/studio.html (accedido en diciembre de 2014).

[3] *@chanchosUY*. *Tránsito del Uruguay (@chanchosUY) | Twitter*. URL: https://twitter.com/chanchosUY (accedido en diciembre de 2014).

[4] Google. *API de rutas de Google*. URL: https://developers.google.com/maps/documentation/directions/ (accedido en diciembre de 2014).

[5] Intendencia de Montevideo. *Cruces de calles correspondientes al departamento de Montevideo*. URL: https://catalogodatos.gub.uy/dataset/cruces-de-calles-montevideo/resource/ab8ed44a-bd5f-40af-bbf1-e781a9336982 (accedido en diciembre de 2014).

[6] Intendencia de Montevideo. *Ubicaciones notables (monumentos) de Montevideo*. URL: https://catalogodatos.gub.uy/dataset/ubicaciones-notables-monumentos-de-montevideo (accedido en diciembre de 2014).

[7] Intendencia de Montevideo. *Vías de Montevideo con cabezales de numeración, significado, tipo y título*. URL: https://catalogodatos.gub.uy/dataset/vias-montevideo-con-cabezales-numeracion-significado-tipo-titulo (accedido en diciembre de 2014).

[8] Yukihiro Matsumoto. *Lenguaje de Programación Ruby*. URL: https://www.ruby-lang.org/es/ (accedido en diciembre de 2014).

[9] *MongoDB*. *MongoDB*. URL: http://www.mongodb.org/ (accedido en diciembre de 2014).

[10] *Oincs*. *Oincs - Tránsito y seguridad - Aplicaciones Android en Google Play*. URL: https://play.google.com/store/apps/details?id=com.trafficapp (accedido en diciembre de 2014).

[11] *retrofit*. *retrofit*. URL: https://github.com/square/retrofit (accedido en diciembre de 2014).

[12] *Sinatra*. *Sinatra*. URL: http://www.sinatrarb.com/ (accedido en diciembre de 2014).

[13] *tweetstream*. *tweetstream*. URL: https://github.com/tweetstream/tweetstream (accedido en diciembre de 2014).

[14] Twitter. *The Streaming APIs*. URL: https://dev.twitter.com/streaming/overview (accedido en diciembre de 2014).