

Karran Rainoldo Besen

**Análise, seleção e teste de ferramentas para
coleta de dados sobre objetos móveis visando
enriquecimento semântico**

Florianópolis/SC

2017

Karran Rainoldo Besen

Análise, seleção e teste de ferramentas para coleta de dados sobre objetos móveis visando enriquecimento semântico

Trabalho de conclusão de curso submetido à Universidade Federal de Santa Catarina como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Universidade Federal de Santa Catarina – UFSC

Departamento de Informática e Estatística

Orientador: Prof. Dr. Renato Fileto

Florianópolis/SC

2017

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Besen, Karran

Análise, seleção e teste de ferramentas para coleta de dados sobre objetos móveis visando enriquecimento semântico / Karran Besen ; orientador, Renato Fileto, coorientador, Italo Lopes, 2017.

60 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Ciências da Computação, Florianópolis, 2017.

Inclui referências.

1. Ciências da Computação. 2. Análise, seleção e teste de ferramentas para coleta de dados sobre objetos móveis visando enriquecimento semântico. 3. Implementação de ferramenta para coleta de dados sobre objetos móveis no momento de postagens no Twitter. I. Fileto, Renato. II. Lopes, Italo. III. Universidade Federal de Santa Catarina. Graduação em Ciências da Computação. IV. Título.

Karran Rainoldo Besen

Análise, seleção e teste de ferramentas para coleta de dados sobre objetos móveis visando enriquecimento semântico

Este Trabalho de Conclusão de Curso foi julgado aprovado para a obtenção do Título de “Bacharel em Ciências da Computação”, e aprovado em sua forma final pela Universidade Federal de Santa Catarina.

Florianópolis, 2017

Prof. Dr. Rafael LCancian
Coordenador
Universidade Federal de Santa Catarina

Banca Examinadora:

Prof. Dr. Renato Fileto
Orientador
Universidade Federal de Santa Catarina

Prof. Dr. Sílvia Modesto Nassar
Universidade Federal de Santa Catarina

Prof. Dr. Ronaldo dos Santos Mello
Universidade Federal de Santa Catarina

Agradecimentos

Agradeço principalmente a minha noiva, Suelen Odraizola, que me motivou e apoiou em todos os momentos. Sempre ajudando a manter o foco e priorizar o término da minha graduação. Não conseguiria finalizar este trabalho sem a ajuda dela.

Agradeço ao meu professor e orientador Renato Fileto que foi extremamente paciente e atencioso comigo e com meu trabalho, me ajudando inclusive enquanto estava na Alemanha. Agradeço ao meu co-orientador Italo Lopes Oliveira que até mesmo em dias atarefados dedicou seu tempo para me orientar.

Agradeço a minha irmã, Cecília Besen, e ao seu namorado, Roque Lemos, que testaram e utilizaram o aplicativo desenvolvido neste trabalho, e agradeço a minha mãe, Vaneide Correia, por todo o carinho e apoio durante este processo.

Por fim, agradeço a todos os meus amigos, professores e colegas que contribuíram com o meu aprendizado e tornaram a faculdade um ambiente agradável.

Resumo

A quantidade de dados gerados pelo amplo uso das mídias sociais (e.g. Twitter, Facebook) é massiva e aumenta a cada segundo. Muitos desses dados estão publicamente disponíveis e podem alimentar uma ampla variedade de aplicações. Entretanto, postagens em mídias sociais têm conteúdo textual não estruturado, sujeito a ruídos e problemas de interpretação. Assim, tais postagens precisam ser semanticamente enriquecidas antes de serem utilizadas em certas aplicações. O processo de enriquecimento semântico de postagens em mídias sociais requer informações sobre o contexto dentro do qual tais postagens são feitas, de modo a gerar anotações de qualidade. Este trabalho apresenta um esforço de coleta e organização de dados para experimentos de enriquecimento semântico de postagens em mídias sociais, uma revisão de literatura sobre ferramentas para coleta de informações adicionais de contexto que possam auxiliar no enriquecimento semântico e uma proposta de aplicativo para coleta de dados sobre objetos móveis no momento de postagens no Twitter. Os resultados obtidos neste trabalho são: (i) um esquema remodelado e expandido para organizar a base de dados sobre objetos móveis, suas postagens e experimentos de enriquecimento semântico de tais postagens no Laboratório de Integração de Sistemas e Aplicações (LISA); (ii) coleta e organização de um grande volume de dados do Twitter, incluindo centenas de milhões de tweets e informações de perfil dos usuários que os postaram; (iii) uma análise comparativa de ferramentas para coleta de informações de contexto de objetos móveis, as quais podem complementar informação de perfil e auxiliar no processo de enriquecimento semântico de postagens em mídias sociais; (iv) uma ferramenta para coleta de dados sobre contexto de objetos móveis no momento de postagens no Twitter; (v) resultados considerando os testes preliminares de anotação semântica dos tweets coletados com dados ligados abertos.

Palavras-chaves: Mídias Sociais. Enriquecimento Semântico. Anotação Semântica. Coleta de Contexto de Objetos Móveis.

Abstract

The amount of data generated by the widespread use of social media (e.g. Twitter, Facebook) is massive and increases every second. Many of these data are publicly available and can feed a wide variety of applications. However, posts in social media have unstructured textual content, subject to noise and interpretation problems. Thus, such posts need to be semantically enriched before being used in certain applications. The process of semantic enrichment of postings in social media requires information about the context within which such posts are made, to generate quality annotations. This work presents an effort to collect and organize data for semantic enrichment experiments of social media postings, a literature review on tools to collect additional context information that may help the semantic enrichment and an application proposal for data collection on mobile objects at the time of Twitter posts. The results obtained in this work are: (i) a remodeled and expanded scheme to organize the database of mobile objects, their posts and experiments of semantic enrichment of such postings in the Laboratory of Integration of Systems and Application (LISA); (ii) collecting and organizing large amounts of Twitter data, including hundreds of millions of tweets and profile information from users who posted them; (iii) a comparative analysis of tools for collecting context information of mobile objects, which can complement profile information and assist in the process of semantic enrichment in social media posts; (iv) a tool for collecting data about the context of mobile objects at the time of Twitter posts; (v) preliminary semantic annotation experiments of tweets collected with open linked data.

Key-words: Social medias. Semantic Enrichment. Semantic Annotation. Context Collection of Mobile Objects

Lista de ilustrações

Figura 1 – Tweet coletado através do script SeMovGet tweet	19
Figura 2 – Anotação semântica para que o tweet da figura 1 possa ser interpretado da forma correta.	19
Figura 3 – Esquema antigo do banco de dados sobre dados coletados do <i>twitter</i>	22
Figura 4 – Esquema do banco de dados sobre objetos móveis e experimentos de anotação semântica	24
Figura 5 – Tela do ODK Build, retirado de (HARTUNG et al., 2010)	27
Figura 6 – Tela inicial do ODK Collect	28
Figura 7 – Arquitetura do sistema ODK 2.0	29
Figura 8 – Tela inicial do GeoODK Collect	30
Figura 9 – Tela do Mobile Data Converter após a conversão dos dados coletados	30
Figura 10 – Tela inicial do KoBo Collect	31
Figura 11 – Arquitetura do gMission	32
Figura 12 – Arquitetura do LoCCAM	34
Figura 13 – Arquitetura do AWARE	35
Figura 14 – Telas de ativação do serviço de acessibilidade do framework AWARE.	39
Figura 15 – Tela do aplicativo desenvolvido	40
Figura 16 – Representação das coordenadas utilizado pelo acelerômetro	41
Figura 17 – Representação das coordenadas utilizado pela rotação	41
Figura 18 – Diagrama de atividades do processo de coleta e carga dos dados	48
Figura 19 – Gráfico comparando a quantidade de Tweets com e sem coordenadas geográficas	48
Figura 20 – Área modificada do modelo do banco de dados do capítulo 3	49
Figura 21 – Tweet utilizado para exemplificar a utilidade do Tweet Context Collector	50
Figura 22 – Mapa gerado pela API do Google Maps com um polígono mostrando a área de Florianópolis definido pelo Twitter	50
Figura 23 – Mapa gerado pela API do google maps com o raio aproximado da localização do objeto móvel.	51
Figura 24 – Mapa gerado pela API do google maps com a orientação do usuário	51
Figura 25 – Tweet utilizado para o teste	52
Figura 26 – Tweet utilizado para o teste	52
Figura 27 – Algoritmo proposto em (FILETO et al., 2015)	53
Figura 28 – Distância calculada pela API do Google Maps	54

Figura 29 – Coordenada e orientação do usuário no momento da postagem do Tweet 55

Lista de tabelas

Tabela 1 – Alguns dos sensores presentes no núcleo do AWARE	36
Tabela 2 – Comparação de ferramentas para coleta de dados sobre objetos móveis	37
Tabela 3 – Comparação dos campos e sensores de entrada presentes nas ferramentas estudadas	38
Tabela 4 – Os campos coletados do acelerômetro	42
Tabela 5 – Os campos coletados do barômetro	42
Tabela 6 – Os campos coletados pela gravidade	43
Tabela 7 – Os campos coletados pelo giroscópio	43
Tabela 8 – Campos coletados pela localização	43
Tabela 9 – Campos coletados pela luminosidade	44
Tabela 10 – Campos coletados pelo magnetômetro	44
Tabela 11 – Campos coletados pela rotação	45
Tabela 12 – Campos coletados pela temperatura	45
Tabela 13 – Campos coletados pelo plugin Google Fused Location	46
Tabela 14 – Campos coletados pelo Google Activity Recognition	46

Lista de abreviaturas e siglas

LOD	Linked Open Data
URI	Uniform Resource Identifier
RDF	Resource description Framework
ODK	Open Data Kit
LISA	Laboratory for Integration of Systems and Applications
PoI	Place of Interest
XML	eXtensible Markup Language
SMS	Short Message Service
CSV	Comma Separated Values

Sumário

1	Introdução	13
1.1	Objetivos	14
1.2	Metodologia	15
1.3	Organização do restante do trabalho	15
2	Fundamentos	16
2.1	Dados sobre objetos móveis	16
2.1.1	Postagens em mídias sociais	16
2.1.2	Dados sobre movimentos	17
2.2	Dados de contexto	17
2.3	Anotação Semântica	18
2.4	Dados Ligados Abertos	19
3	Banco de dados para anotação semântica	21
3.1	Objetivos	21
3.2	Esquema do banco de dados	22
4	Ferramentas para coleta de contextos	25
4.1	Open Data Kit (ODK)	25
4.1.1	ODK Build	26
4.1.2	ODK Aggregate	27
4.1.3	ODK Collect	27
4.1.4	ODK 2.0	28
4.2	GeoODK	29
4.2.1	As diferenças entre GeoODK e ODK	29
4.3	KoBoToolbox	31
4.3.1	As diferenças entre KoBoToolbox e ODK	31
4.4	gMission	32
4.5	LoCCAM	33
4.6	AWARE	34
4.7	Síntese Comparativa	36
5	Aplicativo desenvolvido: Tweet Context Collector	39
6	Resultados	47
6.1	Coleta e análise de dados do Twitter	47
6.1.1	Caracterização dos dados coletados	48

6.2	Testes com ferramentas para coleta de contextos	49
6.2.1	Teste de enriquecimento semântico com LOD	51
7	Conclusões e trabalhos futuros	56
	Referências	58
APÊNDICE A	Artigo sobre o TCC	61
APÊNDICE B	Modelo do banco de dados para experimentos de enriqueci- mento semântico	76
APÊNDICE C	Código fonte Tweet Context Collector	92

1 Introdução

O uso de objetos móveis para a coleta de dados não é novidade, notebooks e palm-tops já são usados a muito tempo para coleta de dados, como por exemplo na saúde (GARRITY; EMAM, 2006), na agricultura (OLIVEIRA; BOLLIGER; FLORIDO, 2006), etc. A redução do custo de aquisição e operação e aumento de funcionalidades de smartphones tornou estes objetos móveis disponíveis até em regiões extremamente pobres e remotas, motiva ainda mais a exploração do uso destes dispositivos na coleta de dados.

Até os celulares mais simples podem ser utilizados na coleta de dados, por exemplo com o uso de serviços de mensagem de texto (e.g. SMS) ou voz. Os smartphones podem servir de formas muito mais sofisticadas para estas coletas, como por exemplo apresentando formulários que possuem interfaces amigáveis, utilizando os dados gerados pelos sensores do dispositivo ou então os dados gerados pela interação com mídias sociais.

A quantidade de conteúdo gerado pelas mídias sociais (e.g. Twitter, Facebook, Instagram, Foursquare) aumenta constantemente. Nas mídias sociais os usuários se tornam fontes de dados. O Twitter, por exemplo, possui atualmente cerca de 310 milhões de usuários ativos¹, i.e., que enviam tweets mensalmente. Esses usuários muitas vezes relatam informações e expressam suas opiniões e sentimentos através de tais tweets, que ficam em grande parte disponíveis via diversas interfaces, incluindo APIs² e frameworks de coleta de dados (KUMAR; MORSTATTER; LIU, 2014). Assim, mídias sociais têm se tornado imensas e ricas fontes de informação para diversas aplicações.

Os dados das mídias sociais são utilizados em uma grande diversidade de pesquisas, tais como recuperação e análise de informação (FRUTUOSO, 2014), análise de sentimentos (KOULOUMPIS; WILSON; MOORE, 2011; AGARWAL et al., 2011) e recomendação (MATHIOUDAKIS; KOUDAS, 2010). As informações adquiridas das mídias sociais podem ser úteis em vários domínios de aplicação, tais como marketing, turismo e segurança pública (ACHREKAR et al., 2011; PAUL; DREDZE, 2011).

Entretanto, dados de mídias sociais apresentam uma enorme variedade de formatos, podendo incluir dados semiestruturados (e.g., tags, timestamps, coordenadas geográficas) e não estruturados (textos livres). Os textos postados em mídias sociais normalmente são informais, sujeitos a muitos erros ortográficos e gramaticais, gírias e acrônimos (e.g. LOL - Lots of laughs, sqn - Só que não, kd - cadê, vc - você), entre outros problemas.

Devido à estrutura linguística irregular, postagens podem ser interpretadas de forma errônea ou com pouca precisão. Assim, para se extrair informação de qualidade,

¹ <https://about.twitter.com/company>

² <https://dev.twitter.com/overview/documentation>

torna-se necessário enriquecer as postagens com referências semânticas bem definidas antes de utilizá-las. A combinação de dados de contexto dos objetos móveis (e.g., informações de perfil) e com os contextos e conteúdos das postagens em mídias sociais tem potencial para produzir anotações semânticas de qualidade, de modo a permitir a interpretação correta dos dados. Com estas anotações semânticas pode-se, por exemplo, desambiguar e identificar recursos para investigar melhor o motivo da viagem de um grupo de pessoas para uma localização em um determinado horário (e.g., Ir a um festival de música, assistir a uma partida de futebol)(WU et al., 2015).

Tendo em vista que o processo de enriquecimento semântico é essencial para a aplicação dos dados coletados de mídias sociais e que informações de contexto podem auxiliar neste processo, este trabalho propõe a coleta e organização de dados para experimentos de enriquecimento semântico de postagens de mídias sociais, além de uma revisão da literatura sobre ferramentas para coleta de informações adicionais de contexto que possam aprimorar o enriquecimento semântico.

1.1 Objetivos

O objetivo geral deste trabalho, é coletar dados de objetos móveis e de mídias sociais, e fazer uma análise comparativa de ferramentas para coleta de informações adicionais sobre os contextos desses objetos, que podem ser, entre outras possibilidades, usuários de mídias sociais e de aplicativos que coletam trajetórias.

Os objetivos específicos deste trabalho são:

- revisão do estado da arte em técnicas e ferramentas para coleta de dados de objetos móveis, incluindo sua localização ao longo do tempo, dados textuais (e.g., postagens em mídias sociais) e informações de contexto;
- continuidade de esforços de coleta e organização de uma coleção de tweets visando experimentos de enriquecimento semântico;
- comparação de ferramentas para coleta de dados de contexto em dispositivos móveis;
- implementação de um aplicativo para coleta de dados sobre dispositivos móveis no momento de postagens no Twitter;
- teste preliminar caracterizando dados coletados do Twitter e o uso do aplicativo para associar coordenadas geográficas a tweets que não as incluíam.

O apoio aos esforços de coleta e organização de dados para experimentos constitui parcela de contribuição adicional, usualmente realizada por cada aluno do nosso grupo para manter dados para experimentos a disposição de todo e de nossos colaboradores.

As contribuições deste trabalho focam principalmente na revisão, seleção e teste de ferramentas para coleta de informações de contexto em dispositivos móveis.

1.2 Metodologia

O desenvolvimento deste trabalho é dividido nas seguintes fases:

1. revisão de técnicas e ferramentas para coleta de dados sobre objetos móveis, incluindo trajetórias e postagens em mídias sociais, além dos formatos dos dados que podem ser coletados;
2. revisão da literatura sobre técnicas e ferramentas para coleta de contextos de objetos móveis visando enriquecimento semântico;
3. busca, seleção, teste e análise comparativa entre as ferramentas de coleta de contexto;
4. Implementação de uma ferramenta para coleta de dados sobre objetos móveis no momento de postagens no Twitter.
5. execução de testes preliminares de enriquecimento semântico de tweets com dados abertos ligados.

1.3 Organização do restante do trabalho

Este trabalho está dividido em seis capítulos incluindo este que é introdutório. O segundo capítulo apresenta a fundamentação teórica com noções sobre anotação semântica, coleta de dados e contexto. O terceiro capítulo apresenta a remodelagem de um banco de dados para enriquecimento semântico. O quarto capítulo realiza uma análise das ferramentas para coleta de dados de contexto de objetos móveis. O quinto apresenta testes realizados com a ferramenta da coleta e por fim o sexto capítulo apresenta a conclusão do trabalho e pesquisas futuras.

2 Fundamentos

Este capítulo apresenta os fundamentos necessários ao entendimento do restante do trabalho. Primeiramente, ele apresenta os conceitos fundamentais de objetos móveis e dados de postagens em mídias sociais e dados sobre movimentos. Posteriormente, ele define informação de contexto dos objetos móveis, incluindo perfis desses objetos e informações sobre o ambiente físico onde se movimentam e os sistemas de informação que utilizam. Finalmente, este capítulo descreve anotações semânticas de dados sobre objetos móveis e delinea como informações de contexto podem ser úteis no processo de anotação semântica.

2.1 Dados sobre objetos móveis

Objetos móveis podem ser pessoas, veículos, animais, etc, portando dispositivos computacionais móveis, tais como smartphones, navegadores ou simples dispositivos de monitoramento de movimentos. Tais dispositivos são pequenos o suficiente para serem portáteis e costumam ser equipados com sensores GPS (*Geographic Positioning System*), o que torna possível fornecer serviços baseados em sua localização. Nas últimas décadas, o uso dos dispositivos móveis cresceu muito. Só no Brasil o número de smartphones em uso já chega a 168 milhões¹.

2.1.1 Postagens em mídias sociais

Com o crescimento da utilização dos dispositivos móveis também aumentou a utilização das mídias sociais remotamente. As postagens em mídias sociais são os registros da interação entre o usuário e a mídia. Este registro pode conter várias informações (e.g. timestamp, localização, fotos, tags e texto) e costuma ficar disponível através da API da mídia em que foi postado. Em (NABO et al., 2014) este registro é representado pela quintupla $SMF(MOid, SMFid, SMid, P, c)$, onde:

- $MOid$ é o identificador do objeto móvel;
- $SMFid$ é o identificador da postagem;
- $SMid$ é o identificador da mídia social da postagem;
- P é um ponto espaço-temporal (usualmente opcional);
- c refere-se aos conteúdos da postagem (e.g., tags, imagens, textos).

¹ <http://www1.folha.uol.com.br/mercado/2016/04/1761310-numero-de-smartphones-em-uso-no-brasil-chega-a-168-milhoes-diz-estudo.shtml>

2.1.2 Dados sobre movimentos

Em (FILETO et al., 2015) dados sobre movimentos são coleções de dados que representam sequências temporalmente ordenadas de posições espaço-temporais de objetos móveis, possivelmente associadas a outras informações (textos, imagens, etc.). Esta definição abrange, entre outras coisas, trajetórias de objetos móveis e trilhas georeferenciadas de usuários em mídias sociais.

Tanto trilhas quanto trajetórias são sequências temporalmente ordenadas de posições espaço-temporais de um objeto móvel, possivelmente associadas com informação adicional. A diferença é que trajetórias têm alta densidade de posições espaço-temporais que podem ser colhidas a curtos intervalos de tempo (e.g., cada 5 segundos) ou curtas distâncias percorridas (e.g., cada 3 metros), enquanto trilhas costumam ser esparsas no espaço e no tempo, devido à natureza assíncrona das postagens dos usuários em mídias sociais (o usuário posta quando quer). Assim, trajetórias permitem análise detalhada dos movimentos e paradas dos objetos móveis, enquanto trilhas em mídias sociais costumam ser mais fáceis de conseguir para grandes conjuntos de usuários em longos períodos de tempo ao longo de grandes regiões ou de todo o globo. Neste trabalho, consideraremos a organização de dados e o enriquecimento semântico tanto de trajetórias quanto postagens em mídias sociais, com ênfase especialmente da parte experimental, nessas últimas.

2.2 Dados de contexto

Dados de contexto caracterizam a situação de uma entidade qualquer, que pode ser, por exemplo, um objeto móvel. O termo contexto é um tanto amplo. Diferentes autores definem contexto de maneiras distintas. (DEY; ABOWD, 2000) definem contexto como "Qualquer informação que pode ser usada para caracterizar a situação de uma entidade considerada relevante para a interação entre usuário e aplicação". Para (SUNDARAM; MANI, 2005) "contexto é qualquer conjunto de informações capaz de afetar a comunicação". Em (SCHILIT; ADAMS; WANT, 1994), contexto inclui informação que refere "aonde você está, quem está com você, e quais recursos estão por perto". Finalmente, (BOLCHINI et al., 2009) definem contexto como "um conjunto de variáveis que podem ser de interesse de um agente e que podem influenciar as suas ações".

No escopo deste trabalho utiliza-se a definição de contexto de (CHEN; KOTZ, 2000), segundo a qual contexto é um espaço de quatro dimensões composto de contexto computacional, contexto físico, contexto temporal e contexto de usuário. Tais dimensões são relevantes para o enriquecimento semântico de dados como postagens em mídias sociais e trajetórias de objetos móveis. O contexto computacional inclui a identificação das aplicações que geram tais dados (e.g., qual mídia social, que ferramenta de coleta de trajetórias), suas configurações (e.g., tipo de conta, taxa de amostragem de dados) e o

histórico das interações do objeto móvel com o sistema computacional. O contexto físico inclui dados coletados por sensores (e.g., velocidade, aceleração, luminosidade), entre outras informações (e.g., dados meteorológicos). O contexto temporal inclui informações de tempo (e.g. dia, semana, mês, ano) em que uma atividade foi realizada. Por fim, contexto de usuário contém as informações relacionadas ao aspecto individual e social do usuário. Tal definição de contexto inclui informação de perfil de usuário de sistema ou agente, sua localização física, o tempo, informações ambientais relevantes e como um usuário interage com um sistema (e.g., situação atual da suas interações) e até o histórico dessas interações.

Alguns dados de contexto podem ser coletados através de APIs das mídias sociais (e.g., perfis de usuários, posições espaciais, timestamp). Outros dados de contexto podem ser coletados através de outras fontes (e.g., dados de sensores e condições meteorológicas podem ser coletados através de aplicações específicas para coleta de dados de objetos móveis). Estes dados de contexto podem ser úteis para desambiguar anotações semânticas (e.g., explorando os contextos espacial e temporal) e constituir regras ou regras para experimentos de enriquecimento semântico.

2.3 Anotação Semântica

Uma anotação semântica associa uma porção de dados (e.g., um trecho de uma trajetória ou trilha, uma parada, uma postagem, um termo relevante do texto de uma postagem) à sua definição precisa e semanticamente bem definida contida, por exemplo, em uma base de conhecimento (e.g., ontologia), vocabulário (e.g., WordNet², VerbNet³) ou conjunto de dados ligados (e.g., DBpedia⁴, LinkedGeoData⁵). Anotações semânticas são úteis para a interpretação correta dos dados, buscas e uma variedade de tarefas e aplicações, pois permitem desambiguar e expandir semanticamente conteúdos de dados não estruturados e semanticamente mal definidos, tais como textos e imagens.

O processo de anotação semântica (geração de anotações semânticas) parte da análise do conteúdo e pode ser viabilizado ou aperfeiçoado de maneira automática mediante o uso de dados sobre o contexto dos objetos móveis, tais como perfis de usuários e dados capturados por sensores acoplados a dispositivos móveis. A figura 1 exibe uma postagem⁶ coletada do Twitter. A interpretação dessa postagem pode ser errônea sem a informação de contexto de que tal postagem foi feita de um estádio durante uma partida de futebol. A figura 2 mostra a anotação semântica desta postagem para evitar interpretações errôneas. Note que o "Porco" mencionado na postagem se refere a um apelido do time de futebol Palmeiras. Além disso, anotações semânticas precisas usando recursos de

² <https://wordnet.princeton.edu/>

³ <http://verbs.colorado.edu/mpalmer/projects/verbnet.html>

⁴ <http://wiki.dbpedia.org/>

⁵ <http://linkedgeodata.org>

⁶ Disponível em <https://twitter.com/wandomoreira/status/607668304447184896>

E vamos para mais uma batalha, vamo assa o Porco, hahahaha... @ Estádio Orlando Scarpelli

View translation

6:59 PM - 7 Jun 2015

Florianópolis, Brazil



Figura 1 – Tweet coletado através do script SeMovGet tweet

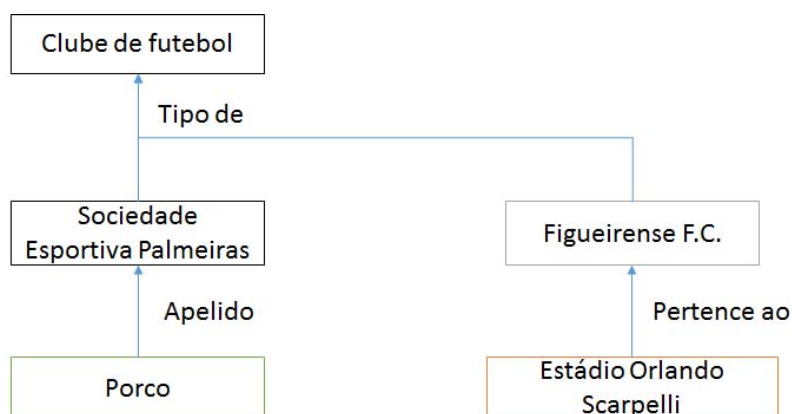


Figura 2 – Anotação semântica para que o tweet da figura 1 possa ser interpretado da forma correta.

dados ligados, permite a expansão semântica. Por exemplo, descobrir facilmente que tanto Palmeiras quanto Figueirense são times de futebol e que o Orlando Scarpelli é o estádio do Figueirense. Daí várias outras informações podem ser recuperadas ou deduzidas.

2.4 Dados Ligados Abertos

Dados ligados abertos, em inglês *Linked Open data (LOD)*, são coleções de dados e conhecimento a eles associados (e.g., classes e suas relações possíveis) oriundas de fontes variadas, que são organizadas e publicadas na Web segundo certas diretrizes⁷ que promovem seu intercâmbio, integração e reuso (NGOMO et al., 2014). A base para a representação de LOD é o modelo RDF (*Resource description Framework*), que descreve tanto dados quanto conhecimento usando a construção básica de tripla rdf, da forma (r, pr, v) , onde:

- r é qualquer recurso na Web identificado unicamente por uma URI;
- pr é uma propriedade do recurso r ;
- v é o valor de tal propriedade, que pode ser um literal ou a URI de um outro recurso.

⁷ <http://linkeddata.org>

Entre as coleções de LOD atualmente disponíveis na Web, DBpedia ([LEHMANN et al., 2015](#)) e LinkedGeoData ([STADLER et al., 2012](#)), têm sido muito usadas para o enriquecimento semântico de dados em geral e de dados sobre movimentos, respectivamente. Conjuntos de dados ligados abertos podem ser consultados, por exemplo, utilizando a linguagem SPARQL⁸, que é uma linguagem dedicada à consulta e manipulação de RDF.

⁸ <https://www.w3.org/TR/sparql11-query/>

3 Banco de dados para anotação semântica

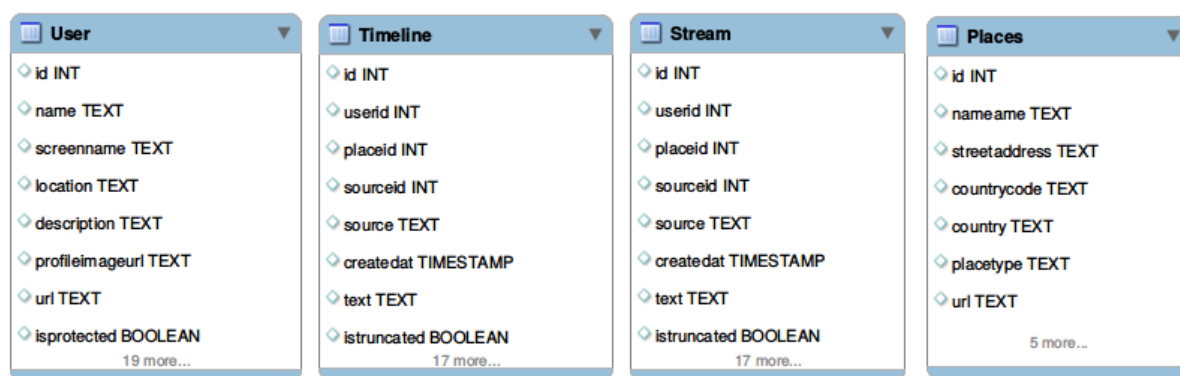
Esta seção apresenta a estrutura geral do banco de dados desenvolvido no escopo deste trabalho para armazenar dados sobre movimento e de mídias sociais, informações sobre os objetos que geraram tais dados, dados para o seu enriquecimento semântico e experimentos. Tal banco de dados é uma evolução de versão anterior mantida no laboratório LISA.

3.1 Objetivos

Os dados gerados por mídias sociais têm componentes semiestruturados (e.g., timestamps, coordenadas opcionais, dados de perfil dos usuários) e não estruturados (e.g., texto livre). Para trabalhar com estes dados é necessário organizá-los em um banco de dados. Por este motivo o banco de dados com o esquema ilustrado na figura 3 era mantido no laboratório LISA (Laboratory for Integration of Systems and Applications). Esta versão possuía quatro tabelas: User, que guardava todos os dados de perfil dos usuários que obtiveram algum tweet coletado; Stream, que continha os dados de tweets coletados em tempo real; Timeline que guardava os dados dos tweets que eram coletados de usuários que estavam na coleção de usuários e usuários mencionados; Places, que guardava os dados de locais relacionados ao tweet. Alguns defeitos deste banco de dados eram:

- Não possuía chaves primárias;
- As tabelas não possuíam relacionamento;
- Não existiam índices;
- As tabelas Timeline e Stream possuíam as mesmas colunas;
- Existiam campos que nunca eram utilizados.

Com o aumento do esforço da coleta de dados para experimentos de enriquecimento semântico, tornou-se necessário a criação de um esquema de banco de dados para organização dos dados coletados. Assim sendo, foi elaborado um esquema para organizar os dados coletados de objetos móveis, de recursos e experimentos de enriquecimento semântico.

Figura 3 – Esquema antigo do banco de dados sobre dados coletados do *twitter*

3.2 Esquema do banco de dados

A fim de organizar dados usados em experimentos de enriquecimento semântico e os próprios resultados desses experimentos no laboratório LISA foi desenvolvido um banco de dado utilizando o modelo relacional. Tal banco pode acomodar dados sobre objetos móveis (incluindo dados de perfil de tais objetos e dados sobre os seus movimentos), informações para o enriquecimento semântico dos dados sobre movimento (que podem ser obtidas de coleções de LOD, por exemplo) e dados dos experimentos de enriquecimento semântico realizados (incluindo responsáveis, métodos e parâmetros utilizados e resultados desses experimentos). O esquema deste banco de dados foi modelado utilizando a ferramenta MySQL Workbench. Tal esquema está ilustrado na figura 4 e é dividido em quatro regiões principais:

- Dados espaço-temporais;
- Dados de contexto espaço-temporais e dos objetos móveis;
- Recursos para enriquecimento semântico;
- Dados sobre experimentos de enriquecimento semântico.

A região de dados espaço-temporais contém os dados sobre movimentos dos objetos móveis. Esta região inclui as fontes (e.g., Twitter, Facebook, Foursquare) e conjuntos de dados que alimentam os dados de movimento. Dados de posições (e.g., Localização de uma pessoa no momento de uma postagem) e segmentos também são incluídos (e.g., posições de duas postagens de um usuário).

A região de dados de contexto espaço-temporais e dos objetos móveis é uma extensão das tabelas de objetos móveis e posições. Esta região contém os dados de perfil do objeto móvel (e.g., Perfil do usuário do twitter), dados de contexto sobre a posição (e.g., Postagem do usuário no twitter), além de informações extraídas da postagem como

conjuntos de palavras relevantes (e.g., Hashtags) e de usuários mencionados, atributos de locais referenciados na postagem (e.g., bairro).

A região de recursos para enriquecimento semântico contém as informações que são utilizados para o enriquecimento semântico e associadas a segmentos e palavras relevantes. Esta região contém os recursos (e.g., Rio 2016, Mineirão) e suas informações: conjunto de geometrias (e.g., Ponto, polígono), fonte (e.g., LinkedGeoData, DBPedia), tipos (e.g., Político, cidade) e os rótulos, que são os textos que referenciam o recurso.

Por fim, a região de dados sobre experimentos de enriquecimento semântico contém informações como período em que o experimento foi realizado, pessoa responsável e métodos e conjuntos de parâmetros utilizados no experimento. Esta região também possui os resultados dos experimentos, podendo ser uma simples extração de palavra relevante ou uma associação de segmento ou palavra relevante à recursos.

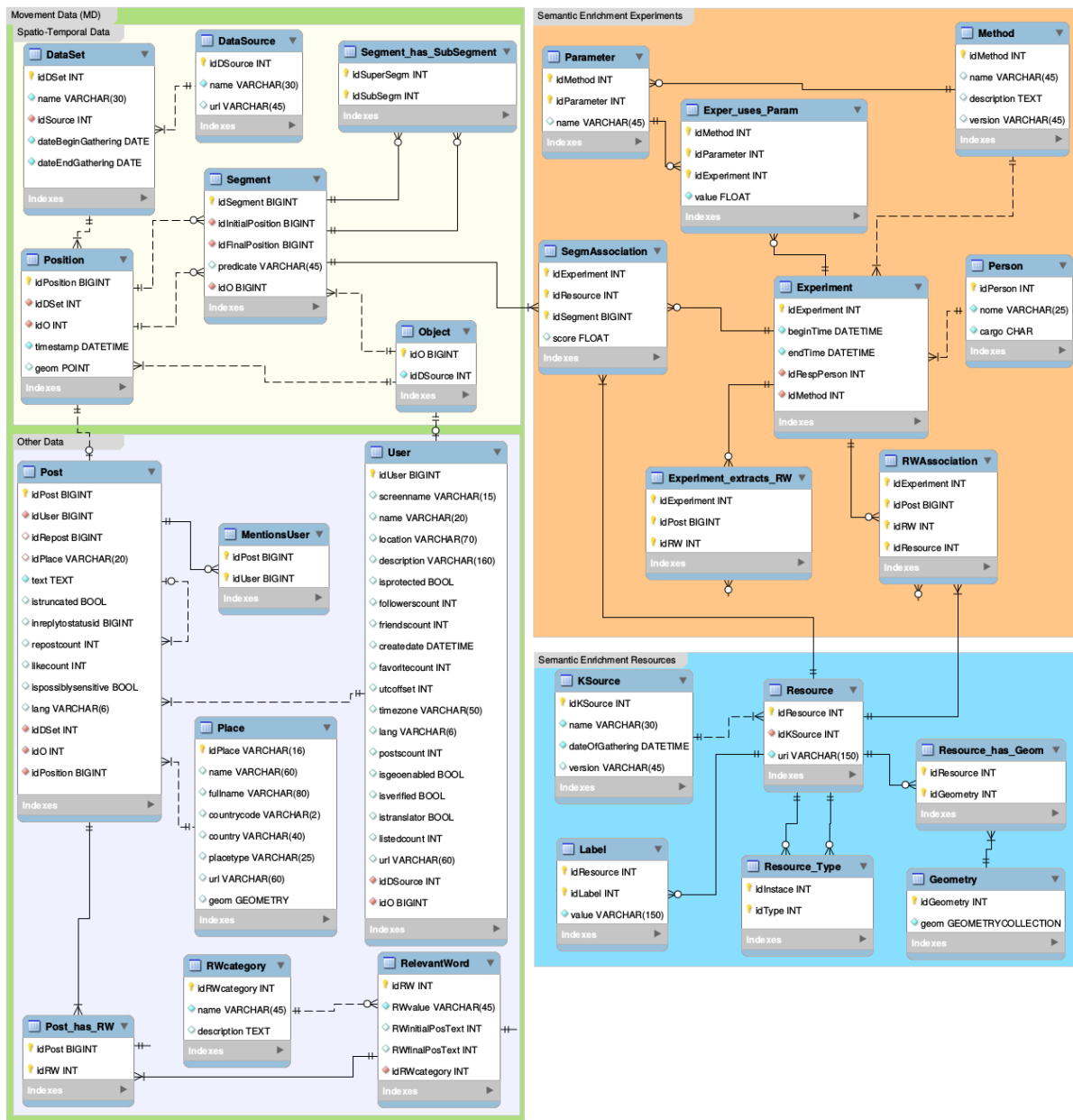


Figura 4 – Esquema do banco de dados sobre objetos móveis e experimentos de anotação semântica

4 Ferramentas para coleta de contextos

Esta seção descreve ferramentas, incluindo bibliotecas, suítes e frameworks, para a coleta de informações de contexto de objetos móveis, de acordo com a documentação encontrada na literatura. Tais ferramentas são apresentadas e comparadas utilizando os seguintes critérios:

- Plataformas suportadas;
- Tipo de licença;
- Entradas: fontes (e.g., sensores), campos;
- Saídas: campos, formatos/APIs;
- Funcionalidades;
- *Burdens* (“Fardos”).

As plataformas suportadas caracterizam os sistemas operacionais que conseguem executar a ferramenta. Os tipos de licença são as definições de autorização de uso da ferramenta. As entradas são as formas em que os dados podem ser coletados pela ferramenta. As saídas são os formatos em que estes dados coletados podem ser recuperados. As funcionalidades são as ações que o programa consegue desempenhar (e.g. Criação de formulários, mapeamento de dados coletados, etc.). Os *Burdens* são as limitações e os requisitos que a ferramenta exige para ser executada.

4.1 Open Data Kit (ODK)

O ODK¹ é uma suíte de ferramentas com o intuito de coletar e gerenciar dados de objetos móveis. Foi desenvolvido pela Universidade de Maryland e pelo Instituto Internacional de análise de sistemas aplicados. A suíte permite a criação de formulários com a ferramenta ODK Build, a hospedagem destes formulários em um servidor com o ODK Aggregate e o preenchimento e envio destes formulários para o servidor através de smartphones com o ODK Collect.

De acordo com (HARTUNG et al., 2010), foi escolhido o padrão *xForms*², para que todas as ferramentas possam ser usadas independentemente, mas podendo interagir entre

¹ <https://opendatakit.org/>

² <http://w3.org/TR/xforms/>

elas. O xForms é um padrão de descrições de formulários desenvolvido pela *World Wide Web consortium (W3C)*³. Esta padronização permite que o ODK importe os formulários de outras ferramentas, como por exemplo o *XLSForm*⁴ e o *koboform*⁵.

Os formulários da suíte ODK suportam as seguintes entradas:

- Texto;
- Inteiro;
- Decimal;
- Alternativa ou múltipla escolha;
- Geopoint (coordenadas GPS);
- Imagens;
- Códigos de barra;
- Timestamp;
- Áudio;
- Vídeo.

4.1.1 ODK Build

A ferramenta ODK Build é uma aplicação web desenvolvida com HTML5 usando Javascript e Ruby Rack. O Build facilita a criação dos formulários com uma interface gráfica drag'n'drop, ilustrada na figura ???. Esta interface gráfica permite a criação dos formulários mesmo sem o conhecimento do padrão XForms. O Build permite que o usuário adicione telas ao formulário com cliques nos botões que representam os tipos de questão. Cada questão tem um conjunto de propriedades que o usuário pode editar. Os usuários também podem reordenar as questões. O Build está hospedado em <http://build.opendatakit.org/> e disponível para download para que os usuários possam utilizá-lo offline em uma máquina local. O Build permite exportar o formulário em XML ou publicar diretamente em um servidor Aggregate.

³ <https://www.w3.org>

⁴ <http://xlsform.org/>

⁵ <http://kobotoolbox.org/>

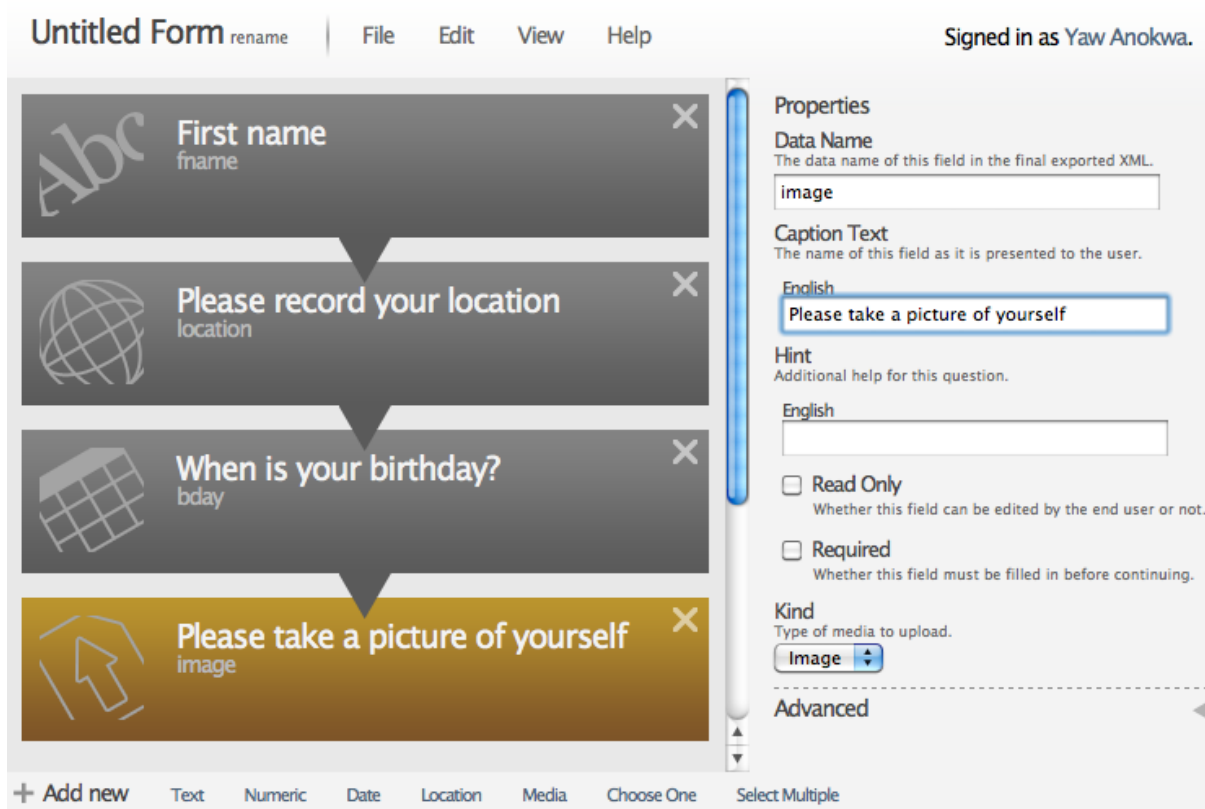


Figura 5 – Tela do ODK Build, retirado de (HARTUNG et al., 2010)

4.1.2 ODK Aggregate

O servidor ODK Aggregate é um repositório que gerência os dados coletados de objetos móveis. O Aggregate foi desenvolvido em Java para rodar em um recipiente web Java. A instalação pode ser feita em um servidor local ou gratuitamente no APP Engine da Google. O Aggregate possui uma interface que permite que o usuário realize consultas sem precisar se preocupar com a linguagem de consulta do banco de dados. O Aggregate pode exportar os dados em vários formatos incluindo CSV (planilhas eletrônicas), KML (Google Earth) e JSON.

4.1.3 ODK Collect

É a parte da suíte dedicada ao usuário que irá responder o formulário. É um aplicativo desenvolvido em Java para objetos móveis que rodem o sistema operacional Android. Como ilustrado na figura 6, o Collect permite que o usuário carregue os formulários salvos no Aggregate, preencha e edite estes formulários carregados e por fim que publique os formulários que foram hospedados no servidor Aggregate. Para suportar operações offline, o Collect armazena os valores dos questionários em arquivos XML e binários (imagens, áudios, vídeos). O usuário pode sincronizar com o Aggregate a qualquer momento utilizando uma conexão com a internet ou transferidos para um computador com um cabo

USB ou cartão de memória.

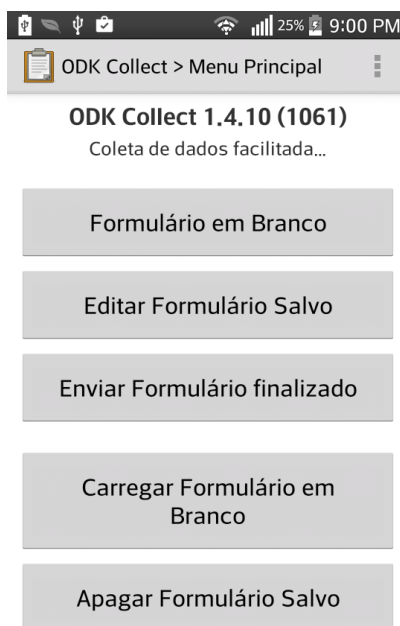


Figura 6 – Tela inicial do ODK Collect

4.1.4 ODK 2.0

Em (BRUNETTE et al., 2013) é dito que uma das funcionalidades que os usuários do ODK mais sentiram falta foi a capacidade de visualizar e analisar os dados através do smartphone. Abaixo estão listadas mais algumas das limitações existentes citadas:

- Dificil navegação entre as questões do formulário;
- Aumentar os tipos de informação que podem ser coletados;
- Falta de outras tecnologias (e.g., papéis, SMS) para coletar os dados.

Pensando nessas limitações, está sendo desenvolvida a versão 2.0 do ODK que ira disponibilizar sincronização bi-direcional (o dispositivo será capaz de recuperar o formulário enviado), a opção de análise dos dados coletados através de smartphones, melhor flexibilidade na navegação dos formulários e a capacidade de tratar formulários respondidos através de papéis e SMS.

A figura 7, retirada do (BRUNETTE et al., 2013) ilustra a arquitetura do ODK 2.0. Na parte esquerda é mostrada a parte dos serviços hospedados na nuvem, aonde o Aggregate fornece serviços para sincronizar dados com os dispositivos e exportar estes dados. Na parte direita é ilustrada a parte dos serviços dos clientes móveis onde são compartilhados um banco de dados e um sistema de arquivo. Scan, Survey e Tables são as

ferramentas para coletar, processar e visualizar os dados de entrada. Submit e Sensors são ferramentas para o android ter os serviços necessários.

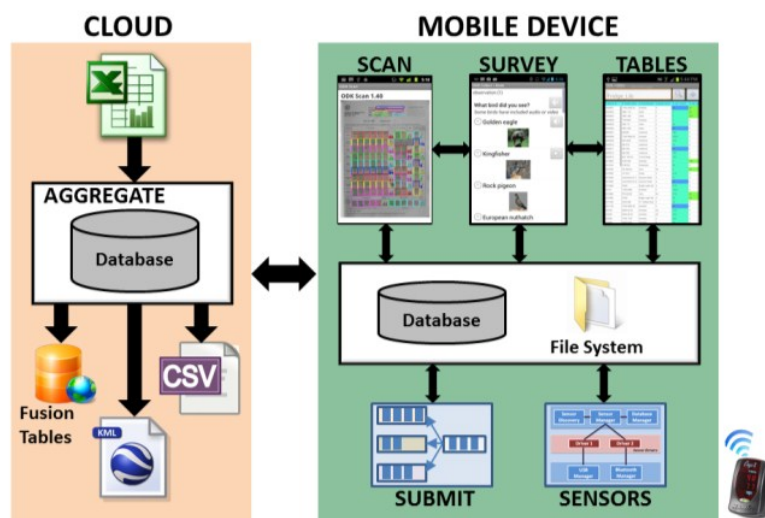


Figura 7 – Arquitetura do sistema ODK 2.0

4.2 GeoODK

A GeoODK⁶ é uma modificação Opensource do ODK feita pelo Instituto Internacional de Análise de Sistemas Aplicados e pela Global Agricultural Monitoring. A GeoODK fornece uma maneira de coletar e armazenar informações geo-referenciadas. Esta suíte também fornece ferramentas para visualização, análise e manipulação dos dados.

4.2.1 As diferenças entre GeoODK e ODK

As suítes GeoODK e ODK possuem os mesmos servidores e ferramentas para criação de formulários. A maior diferença entre as duas suítes são as funcionalidades de mapeamento existentes na GeoODK. Ilustrada na figura 8, a tela inicial do GeoODK Collect é diferente do ODK Collect. Entre as funcionalidades de mapeamento adicionais estão:

- Mapeamento offline;
- Visualizações dos dados coletados no dispositivo móvel;
- Coleta de polígonos e traços do GPS.

Além disto, a GeoODK fornece o Mobile Data Converter, que é um software que permite a conversão de dados coletados para formatos espaciais e a preparar um conjunto

⁶ <http://geoodk.com/>



Figura 8 – Tela inicial do GeoODK Collect

de dados para um sistema GIS. Para isto basta, baixar e selecionar os dados coletados através servidor Aggregate em arquivos CSVs, selecionar os campos referentes à localização e opcionalmente à imagens e por fim selecionar para que formato você deseja que o arquivo seja exportado. A figura 9, retirada da sessão de tutorias da GeoODK ⁷, ilustra a tela do mobile data converter após converter os dados coletados com sucesso.

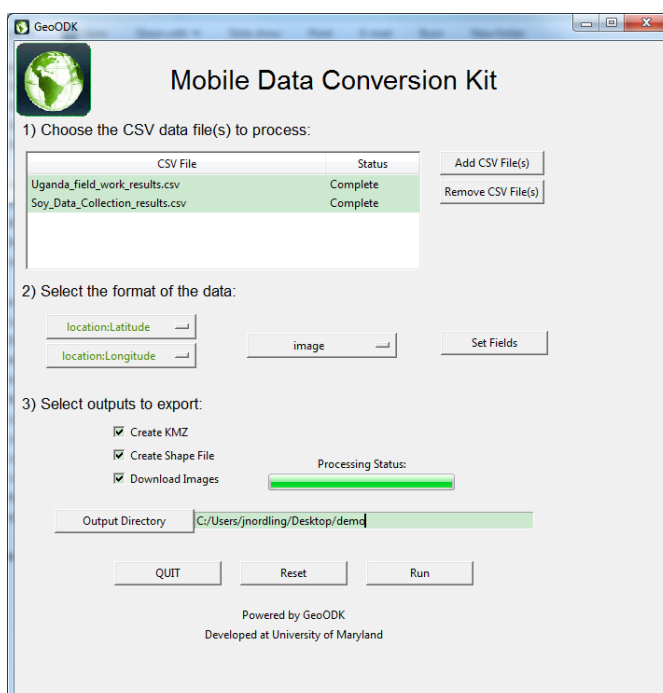


Figura 9 – Tela do Mobile Data Converter após a conversão dos dados coletados

⁷ http://geoodk.com/mdk_howto.php

4.3 KoBoToolbox

A KoBoToolbox⁸ é uma suíte de ferramentas Opensource baseadas no ODK para coleta de dados de objetos móveis para organizações humanitárias e pesquisadores. Foi Desenvolvido pela Harvard Humanitarian Initiative. A suíte é composta pelas ferramentas:

- koboform (dkobo, django kobo) - Ferramenta web para criar os formulários;
- kobocat and kobocat-templates - Servidor para hospedar formulários e analisar os dados;
- enketo-express - Aplicação Web, desenvolvida em HTML para coletar, pré-visualizar formulários e editar os dados submetidos;
- kobocollect - Aplicativo android para coleta de dados.

4.3.1 As diferenças entre KoBoToolbox e ODK

As duas ferramentas suportam formulários criados a partir do XLSForm e servidores Aggregate. Porém, a KoBoToolbox disponibiliza um servidor próprio para o gerenciamento e a coleta de dados. Este servidor possui uma ferramenta própria que facilita o desenvolvimento dos formulários, o koboform. O aplicativo de coleta da KoBoToolbox, o Kobo Collect é visualmente parecido com o ODK Collect, como ilustrado na figura 10, porém já é previamente configurado para a conexão com o servidor da KoBoToolbox. Por padrão cada organização que utiliza o servidor da KoBoToolbox tem direito a 10 mil envios de formulários e 5 GB do banco de dados, porém é possível criar um servidor local ou entrar em contato com a equipe KoBoToolbox para solicitar mais recursos.

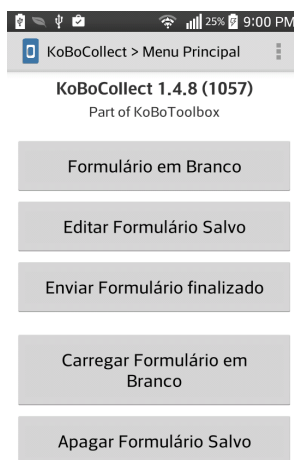


Figura 10 – Tela inicial do KoBo Collect

⁸ <http://kobotoolbox.org/>

4.4 gMission

O aplicativo gMission foi desenvolvido pela Hong Kong University of Science and Technology e tem como objetivo crowdsourcing espacial. O gMission é open source e está disponível para Android e iOS. Existe pouca documentação e usuários que utilizam o gMission.

Em resumo, o sistema permite que os usuários realizem perguntas para usuários que se encontrem em um determinado local. O gMission conta com um sistema de contribuição explicado em (CHITILAPPILLY; CHEN; AMER-YAHIA, 2016). Nele para quanto mais tarefas forem cumpridas, mais pontos são adquiridos. Quanto mais pontos uma pessoa tiver, mais perguntas podem ser criadas.

Como ilustrado na figura 11 retirado de (CHEN et al., 2010), o gMission possui a interface de usuário que é responsável por fornecer ao usuário as funções de postar e responder tarefas, e o gerenciador de funções que possui três módulos principais:

- **Sensor de localização:** Responsável pela localização do usuário, baseado no GPS e no Wi-Fi;
- **Recomendação:** Recomenda as tarefas a partir das informações geográficas e da quantidade de tarefas;
- **Controle de qualidade:** Responsável pela deleção das tarefas que ultrapassarem um certo prazo, pelo sistema de contribuição e não permite respostas de localizações erradas.

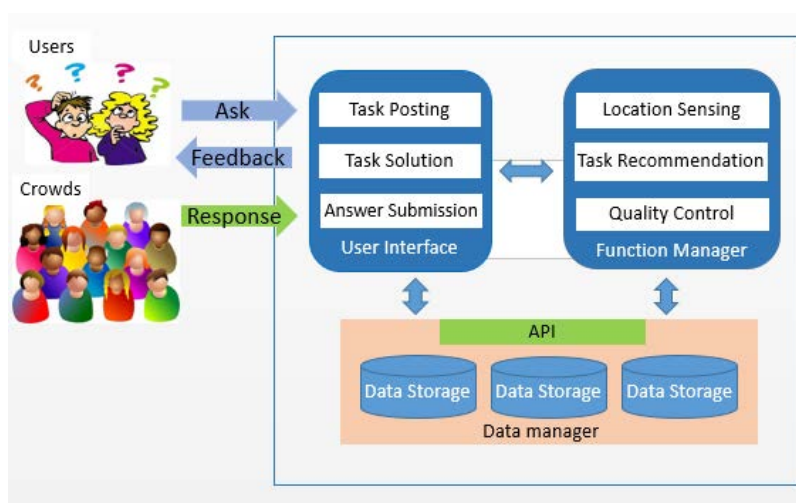


Figura 11 – Arquitetura do gMission

4.5 LoCCAM

O LoCCAM⁹ é um middleware para sistemas móveis e sensíveis ao contexto desenvolvido pelo Grupo de Redes de Computadores e pela Engenharia de Software e Sistemas (GREat) da Universidade Federal do Ceará (UFC)(DUARTE et al., 2015). Com o LoCCAM é possível desenvolver aplicações sensíveis ao contexto rapidamente, como por exemplo uma aplicação com o objetivo de mapear determinados locais com informações referentes à localização do dispositivo¹⁰.

A arquitetura do LoCCAM, ilustrada na figura 12 retirada do site do LoCCAM¹¹, pode ser dividida em duas partes principais (DUARTE et al., 2014):

- **SysSU (System Support for Ubiquity):** Recebe a relação de interesses das aplicações e as repassa para o CAM;
- **CAM (Context Acquisition Manager) Framework :** armazena a relação de interesses e controla os CACs (Componentes de aquisição de contexto).

O SysSU Filter realiza o filtro nas informações dos sensores, o adaptation reasoner cuida da lista de interesses da aplicação e se comunica com o CAC manager que controla todos os CACs. Abaixo estão listados os CACs disponíveis no LoCCAM:

- Temperatura;
- Luminosidade;
- Aceleração;
- Gravidade;
- Campo Magnético;
- Localização GPS;
- Localização Wi-Fi;
- Random;
- Clima;
- Wi-fi.

⁹ <http://loccam.great.ufc.br/>

¹⁰ <http://loccam.great.ufc.br/downloads/download-aplicacoes.html>

¹¹ <http://loccam.great.ufc.br/informacoes-gerais/arquitetura.html>

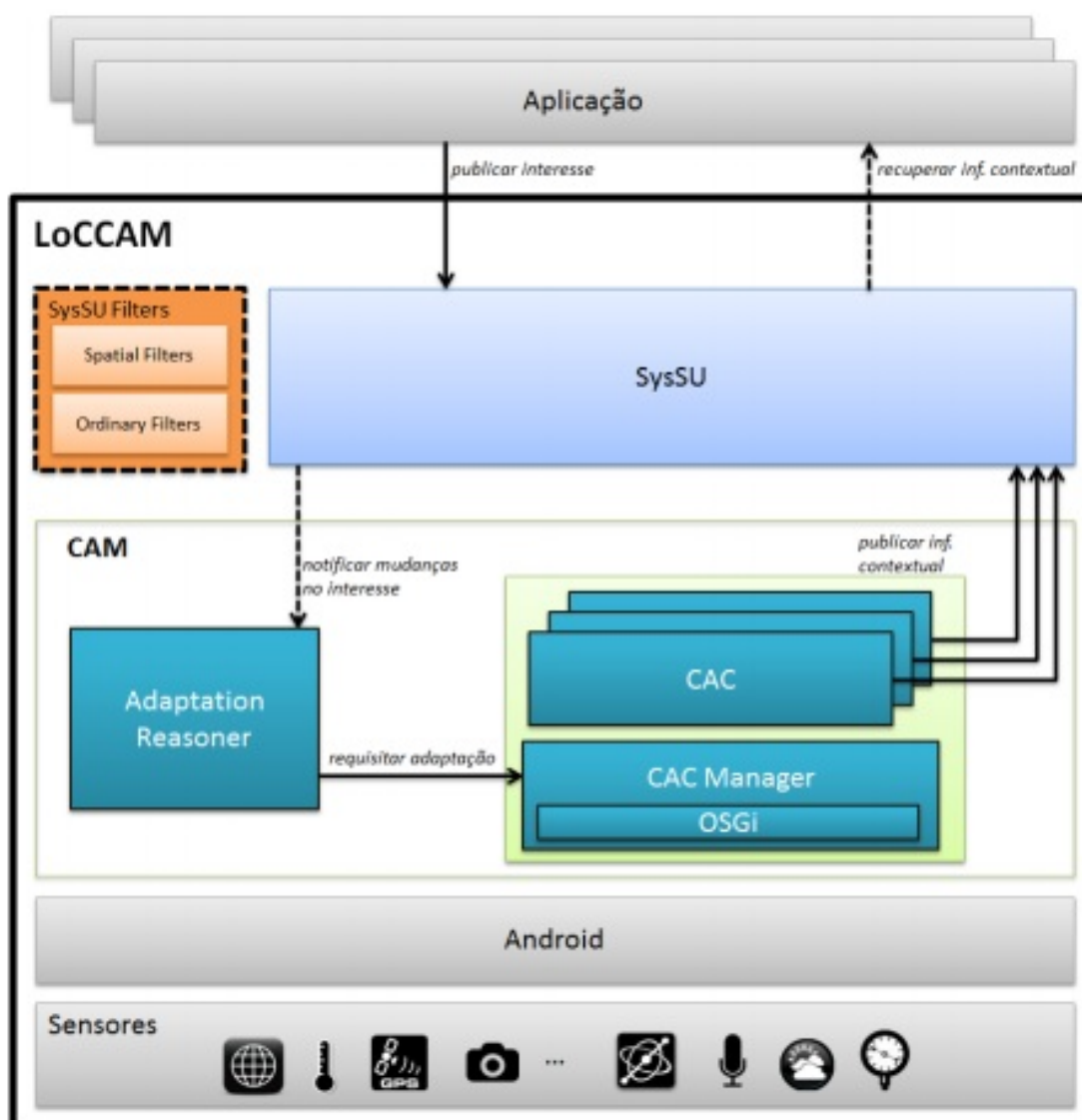


Figura 12 – Arquitetura do LoCCAM

4.6 AWARE

O AWARE¹² é um middleware open-source dedicado a captura de informações de contexto móveis desenvolvido na Carnegie Mellon University. Pode ser utilizado para criar aplicativos para sistemas operacionais Android e iOS. Com ele é possível criar aplicações sensíveis ao contexto e coletar dados de contexto.

A figura (13 retirada de (FERREIRA; KOSTAKOS; DEY, 2014)) ilustra a arquitetura do AWARE. O AWARE utiliza **sensores** como forma de capturar dados de contextos de hardware (e.g., Acelerômetro, giroscópio, magnetismo), software (e.g., calendário, email, status das aplicações) e humanos (e.g., ESM - Experience sampling Method, entradas de voz e gesto). A Tabela 1 retirada de (FERREIRA, 2013) apresenta alguns

¹² <http://www.awareframework.com/>

dos sensores já incluídos no AWARE, também é apresentado o tipo (i.e. HW - Hardware, SW - Software e H - Humano) e uma breve descrição de cada sensor. O AWARE possui **plugins**, que são extensões que permitem os desenvolvedores implementar códigos que utilizam sensores ou até mesmo outros plugins. Um plugin deve refinar o contexto oferecido por algum sensor, fornecendo contextos mais precisos ou mais compreensíveis. Para a comunicação entre sensores, plugins e aplicações são utilizadas as estratégias de **broadcast**, **observador** e **fornecedor**.

- **Broadcast:** atualiza os outros sensores, as aplicações e os plugins com o contexto do usuário;
- **Fornecedor:** armazena os dados dos sensores e do plugin localmente no dispositivo ou se necessário no servidor MySQL;
- **Observador:** monitora as mudanças dos dados dos sensores e dos plugins em tempo real.

Por fim, o dispositivo pode enviar os dados adquiridos dos sensores para outros dispositivos por MQTT (Padrão de telemetria) ou para um banco de dados MySQL por HTTPS. Os dados podem ser armazenados gratuitamente no **AWARE Server**.

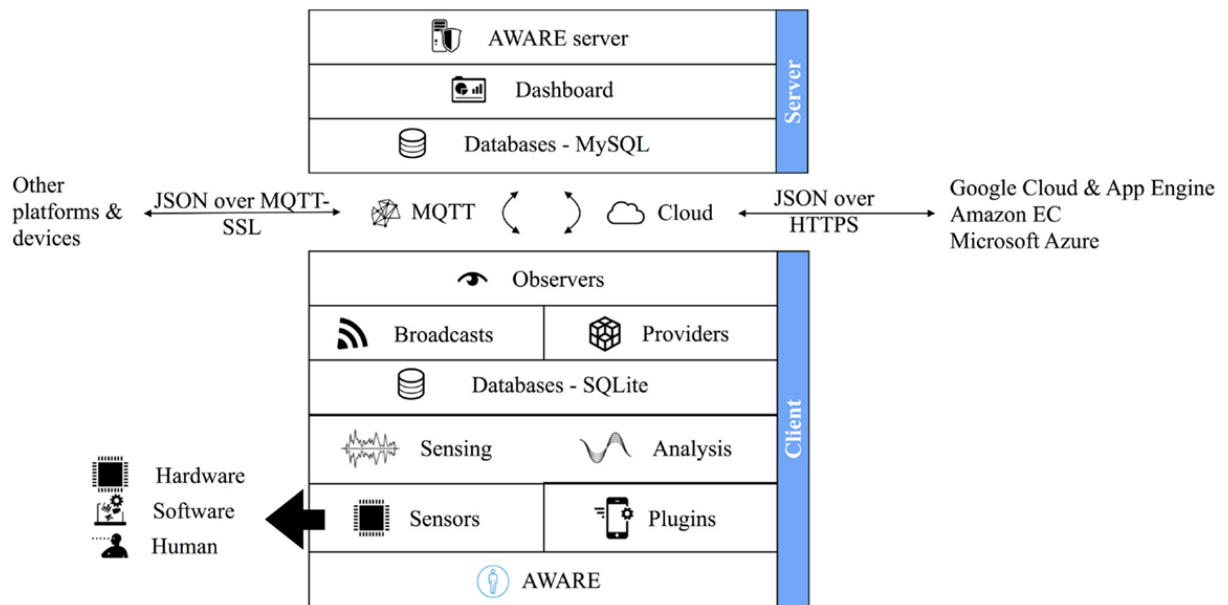


Figura 13 – Arquitetura do AWARE

Tabela 1 – Alguns dos sensores presentes no núcleo do AWARE

Sensores	Tipo	Descrição
Acelerômetro	HW	Sensor da força de aceleração de acordo com o eixo do dispositivo, inclui gravidade (m/s ²)
Aplicações	SW	Aplicações usadas pelo dispositivo
Barômetro	HW	pressão atmosférica (mbar/hPa)
Bateria	HW e SW	Dados de eventos de bateria e energia (e.g. reiniciar, desligar)
Bluetooth	HW	Informações do bluetooth, realiza escaneamentos em intervalos de tempo procurando dispositivos
Comunicação	SW	dados de comunicação do dispositivo (e.g. Dados de ligações, mensagens)
ESM	H	Dados providos pelo usuário a partir de questionários.
Gravidade	HW	Força da gravidade (m/s ²)
Giroscópio	HW	Fator de rotação ao redor do eixo do dispositivo (rad/s)
Instalações	SW	Aplicações adicionadas, removidas e atualizadas do dispositivo
Luminosidade	HW	Luminosidade do ambiente (lux)
Aceleração lineear	HW	Força de aceleração excluindo a gravidade (m/s ²)
Localização	HW e SW	Localização através da rede e/ou GPS do dispositivo móvel.
Magnetismo	HW	Força geomagnética aplicada ao eixo do dispositivo (micro-tesla)
Rede	SW	Uso da rede (e.g. modo avião, Wi-Fi, Rede móvel...)
Orientação	HW	Ângulo de orientação do dispositivo (graus)
Processador	SW	Dados do uso do processador (e.g., processos do SO, processos do usuário, inativo)
Proximidade	HW	Distância entre dispositivo e objeto (próximo/longe ou cm)
Rotação	HW	Vetor de rotação em relação ao eixo do dispositivo
Tela	HW e SW	Status da tela (e.g. Ligado ou desligado) e eventos de travar e destravar
Telefonia	HW	Dados de capacidades de telefonia do dispositivo (e.g. Velocidade da rede, tipo)
Temperatura	HW	Dados de temperatura do ar (Celsius)
Tráfego	SW	Dados de tráfego da rede do Wi-Fi, Bluetooth e rede móvel (e.g., pacotes e bytes)
Wi-Fi	HW	Informações da Wi-Fi e buscas intervaladas de dispositivos vizinhos

4.7 Síntese Comparativa

Nesta seção foram utilizados os critérios do início deste capítulo para realizar uma síntese comparativa das ferramentas estudadas. O objetivo da tabela 2 é comparar os tipos de licença e as plataformas, entradas e saídas suportadas pelas ferramentas estudadas.

Na tabela 3 foram comparadas os campos e os sensores de entradas presentes nas ferramentas estudadas. As ferramentas ODK, GeoODK e KoBoToolbox aceitam os

Tabela 2 – Comparação de ferramentas para coleta de dados sobre objetos móveis

Ferramentas	Plataformas	Licenças	Entradas	Saídas
ODK	Android	Opensource	Formulários	KML, JSON, CSV
GeoODK	Android	Opensource	Formulários	ESRI, KML, JSON, CSV
KoboToolbox	Android, iOS	Opensource	Formulários	KML, JSON, CSV, SPSS
gMission	Android, iOS	Opensource	Formulários	-
LoCCAM	Android	Freeware	Sensores	Tuplas
AWARE	Android, iOS	Opensource	Formulários, Sensores	CSV

mesmos campos e por isso estão listadas na mesma coluna da tabela.

Com as tabelas comparativas é possível notar que das ferramentas estudada que apenas suportam formulários como forma de entrada, apenas a KoboToolbox tem suporte aos sistemas operacionais Android e iOS. Também é possível notar que a única ferramenta estudada não opensource é o middleware LoCCAM.

Ainda com o uso das tabelas é possível identificar o AWARE como a ferramenta mais completa. O AWARE é suportado pelos sistemas operacionais Android e iOS, é a única das ferramentas estudadas com a capacidade da coleta de dados de sensores e formulários, e também é a ferramenta que possui a maior quantidade de sensores já implementados em seu núcleo.

Formato de entrada	Campos	ODK, GeoODK, KoboToolbox	LoCCAM	AWARE
Sensores	Acelerômetro		X	X
	Aplicações			X
	Barômetro			X
	Bateria			X
	Bluetooth			X
	Comunicação			X
	Gravidade		X	X
	Giroscópio			X
	Instalações			X
	Luminosidade		X	X
	Aceleração linear			X
	Localização	X	X	X
	Magnetômetro		X	X
	Rede			X
	Orientação			X
	Processador			X
	Proximidade			X
	Rotação			X
	Tela			X
	Telefonia			X
Temperatura		X	X	
Tráfego			X	
Wi-Fi			X	X
Formulário	Alternativa ou múltipla escolha	X		X
	Decimal	X		
	Inteiro	X		X
	Imagens	X		
	Códigos de barra	X		
	Áudio	X		
	Vídeo	X		
	Texto	X		X

Tabela 3 – Comparação dos campos e sensores de entrada presentes nas ferramentas estudadas

5 Aplicativo desenvolvido: Tweet Context Collector

Por ser a ferramenta estudada mais completa, o AWARE foi utilizado para o desenvolvimento de um aplicativo para a realização de testes de associação de coordenadas geográficas a Tweets que não às possuíam. O aplicativo pode ser executado em sistemas Android e para poder coletar e enviar as informações dos sensores, o aplicativo deve estar pelo menos rodando em segundo plano. Também é necessária a ativação do serviço de acessibilidade do AWARE framework, visível na figura 14.

Como o Twitter lança uma notificação no celular quando o usuário posta um Tweet, foi utilizado o sensor de aplicativos existente no núcleo do AWARE para criar um observador de notificações. Desta forma, o Tweet Context Collector identifica quando o Twitter cria uma notificação, e então ativa os sensores e envia os dados recentemente coletados para um banco de dados MySQL.

O aplicativo possui uma única tela, como mostrado na figura 15. Nesta tela é possível visualizar o identificador único do objeto móvel gerado pelo AWARE. Também é possível visualizar um botão que serve como alternativa para a ativação da função de coleta e envio dos dados dos sensores. Este botão foi adicionado pois em alguns casos os usuários esqueceram de executar o aplicativo e só lembraram depois de postar o Tweet.

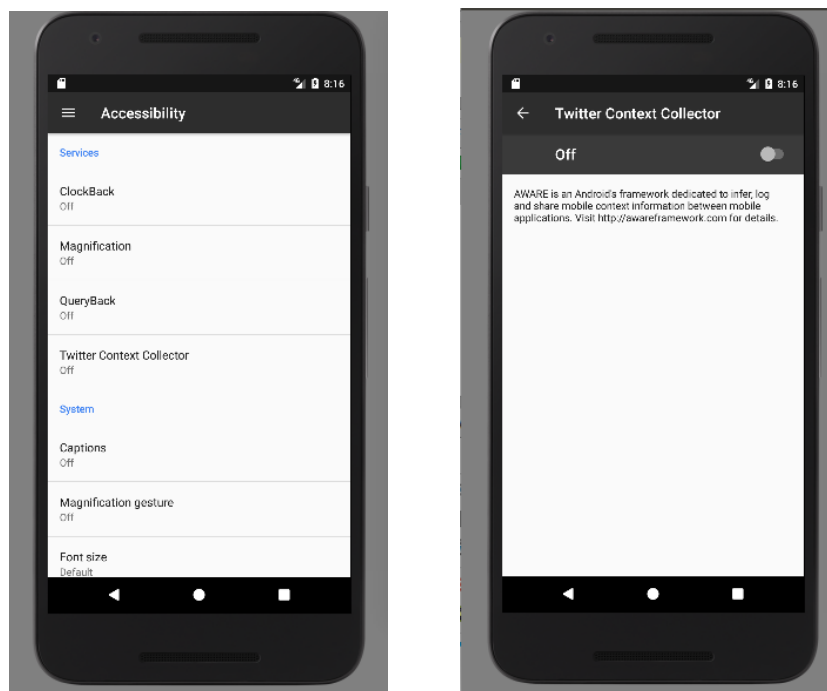


Figura 14 – Telas de ativação do serviço de acessibilidade do framework AWARE.



Figura 15 – Tela do aplicativo desenvolvido

Enquanto o aplicativo está sendo executado, os sensores de baixa frequência ficam ativos. Os sensores de baixa frequência ficam periodicamente gravando localmente os valores mais recentes. Para diminuir o consumo de bateria e a quantidade de registros, os sensores de alta frequência ficam inativos até que a função de coleta dos dados é executada. Quando a função de coleta de dados é executada, cada sensor de alta frequência é ativo por um segundo e os valores coletados são enviados para o servidor. Assim que todos os dados dos sensores são enviados para o servidor, os dados são apagados do objetos móveis.

Em geral, os smartphones possuem diferentes sensores disponíveis. Por exemplo o sensor de temperatura é muito raro de ser encontrado nos celulares, porém ele está implementado e caso o aplicativo seja instalado em um celular com o sensor disponível os dados da temperatura serão coletados.

Cada sensor possui um campo de precisão chamado accuracy. A maioria dos sensores do Tweets Context Collector utiliza o sistema de precisão definido pelo Android¹. Os campo de precisão podem receber valores de 1-3, sendo 3 a precisão mais alta. O sensor location é o único em que o nível de precisão é definida em metros.

¹ <https://developer.android.com/reference/android/hardware/SensorManager.html>

Alguns dos sensores do Tweet Context Collector precisam de um sistema de coordenadas. Na maioria dos sensores, as coordenadas são relativas à tela do objeto móvel, como representado na figura 16, retirada do site do AWARE². O único dos sensores utilizado com um sistema de coordenadas diferente é o sensor de rotação, representado pela figura 17, retirado do site do AWARE³.

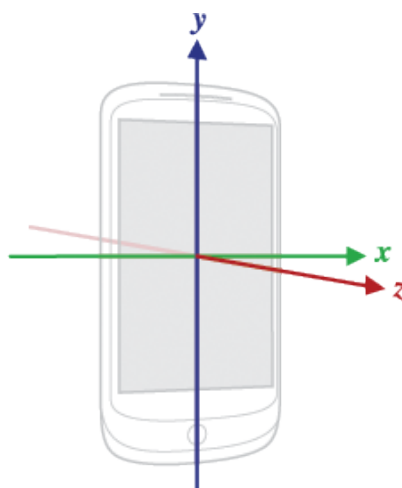


Figura 16 – Representação das coordenadas utilizado pelo acelerômetro

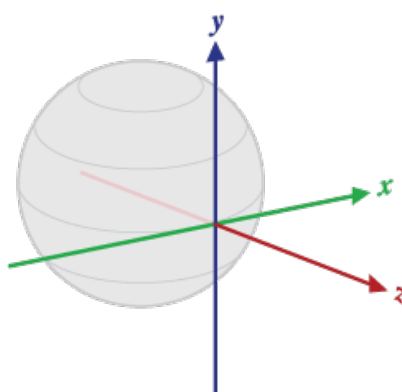


Figura 17 – Representação das coordenadas utilizado pela rotação

² <http://www.awareframework.com/accelerometer/>

³ <http://www.awareframework.com/rotation/>

Os sensores implementados no Tweet Context Collector são:

- Acelerômetro;
- Barômetro;
- Gravidade;
- Giroscópio;
- Localização;
- Luminosidade;
- Magnetômetro;
- Rotação;
- Temperatura.

Acelerômetro O acelerômetro é um sensor de alta frequência que mede a aceleração aplicada ao sensor embutido no dispositivo, incluindo a força da gravidade. Os campos coletados estão representados na tabela 4.

Tabela 4 – Os campos coletados do acelerômetro

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
double_values_0	Real	Valor do eixo X
double_values_1	Real	Valor do eixo Y
double_values_2	Real	Valor do eixo Z

Barômetro O Barômetro é um sensor de alta frequência que mede a pressão do ar no ambiente. O barômetro pode ser utilizado para detectar e prever pequenas mudanças no tempo, por exemplo, uma queda de pressão indica chuva, e o aumento da pressão representa um bom tempo. Os campos estão representados na tabela 5

Tabela 5 – Os campos coletados do barômetro

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
double_values_0	Real	Pressão do ar no ambiente, em mbar/hPa (dependendo do hardware)

Gravidade A gravidade é um sensor de alta frequência que mede a força da gravidade aplicada no sensor imbutido no dispositivo e fornece um vetor tridimensional

indicando a direção e a magnitude da gravidade (em m/s). Quando o dispositivo está parado, tanto a gravidade quanto o acelerômetro devem ter os mesmos valores. Os campos coletados pelo sensor são representados pela tabela 6.

Tabela 6 – Os campos coletados pela gravidade

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
double_values_0	Real	Valor do eixo X
double_values_1	Real	Valor do eixo Y
double_values_2	Real	Valor do eixo Z

Giroscópio O giroscópio é um sensor de alta frequência que mede a rotação em torno do dispositivo em rad/s. A rotação é positiva em sentido anti-horário. Os campos coletados pelo sensor são representados pela tabela 7.

Tabela 7 – Os campos coletados pelo giroscópio

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
double_values_0	Real	Valor do eixo X
double_values_1	Real	Valor do eixo Y
double_values_2	Real	Valor do eixo Z

Localização A localização é um sensor de baixa frequência que fornece a melhor estimativa de localização do dispositivo móvel. a rotação em torno do dispositivo em rad/s. A rotação é positiva em sentido anti-horário. Os campos coletados pelo sensor são representados pela tabela 8.

Tabela 8 – Campos coletados pela localização

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	A precisão estimada em metros
double_longitude	Real	A longitude da localização em graus
double_latitude	Real	A latitude da localização em graus
double_altitude	Real	A altitude da localização, se disponível, em metros sobre o nível do mar
double_bearing	Real	A direção em que a pessoa está andando, em graus
double_speed	Real	A velocidade do dispositivo, se disponível, em m/s
provider	Texto	GPS ou Wi-Fi

Luminosidade A luminosidade é um sensor de alta frequência que mede a luminosidade do ambiente. Pode ser usado para detectar a luminosidade em ambientes fechados ou abertos. Os valores de luminosidade são definidos pelo sistema android ⁴, os constantes oficiais são:

- Cloudy sky: 100.0
- Full moon: 0.25
- No moon: 0.001
- Overcast: 10000.0
- Shade: 20000.0
- Sunlight: 110000.0
- Sunlight maximum: 120000.0
- Sunrise: 400.0

Os campos coletados pelo sensor são representados pela tabela 9

Tabela 9 – Campos coletados pela luminosidade

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
double_light_lux	Real	A luminosidade do ambiente (lux)

Magnetômetro O magnetômetro é um sensor de alta frequência que mede a a força do campo geomagnético ao redor do dispositivo. Este sensor fornece os dados de força (em μT) para cada um dos eixos. Os campos coletados pelo sensor são representados pela tabela 10

Tabela 10 – Campos coletados pelo magnetômetro

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
double_values_0	Real	Valor do eixo X
double_values_1	Real	Valor do eixo Y
double_values_2	Real	Valor do eixo Z

⁴ <https://developer.android.com/reference/android/hardware/SensorManager.html>

Rotação A rotação é um sensor de alta frequência que mede a orientação do dispositivo com uma combinação de um ângulo e um eixo. Os três elementos do vetor de rotação são expressados em:

- $x \cdot \sin(\theta/2)$
- $y \cdot \sin(\theta/2)$
- $z \cdot \sin(\theta/2)$

Os campos coletados pelo sensor são representados pela tabela 11

Tabela 11 – Campos coletados pela rotação

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
double_values_0	Real	O vetor de rotação ao redor do eixo x, $x \cdot \sin(\theta/2)$
double_values_1	Real	O vetor de rotação ao redor do eixo y, $y \cdot \sin(\theta/2)$
double_values_2	Real	O vetor de rotação ao redor do eixo z, $z \cdot \sin(\theta/2)$
double_values_3	Real	$\cos(\theta/2)$ (depende do fabricante)

Temperatura O sensor de temperatura é um sensor que mede a temperatura do ambiente, em celsius ($^{\circ}\text{C}$). Poucos dispositivos possuem este sensor. Os campos deste sensor são representados pela tabela 12

Tabela 12 – Campos coletados pela temperatura

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
accuracy	Inteiro	Nível de precisão do sensor
temperature_celsius	Real	Temperatura medida

O aplicativo também foi integrado aos plugins de baixa frequência Google Activity Recognition e Google Fused Locations.

Google Fused Location O plugin Google Fused Location usa a API do Google Fused Location para fornecer a localização do usuário em uma forma eficiente energeticamente. Além dos campos do sensor de localização, também são coletados os campos da tabela 13

Google Activity Recognition

O plugin Google Activity Recognition detecta o meio de transporte do usuário. Dentre os meios de transportes identificados estão:

- IN_VEHICLE O dispositivo está com um usuário em um veículo automotivo

Tabela 13 – Campos coletados pelo plugin Google Fused Location

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
geofence_label	Real	Nome da área geográfica
double_latitude	Real	latitude da área geográfica, em metros
double_longitude	Real	longitude da área geográfica, em metros
double_radius	Real	Raio da área geográfica, em metros

- ON_BICYCLE O dispositivo está com um usuário em uma bicicleta
- ON_FOOT O dispositivo está com um usuário que está caminhando ou correndo
- RUNNING O dispositivo está com um usuário que está correndo
- STILL O dispositivo está com um usuário que está parado
- TILTING O ângulo do dispositivo em relação a gravidade mudou significativamente
- UNKNOWN Não foi possível identificar a atividade do dispositivo
- WALKING O dispositivo está com um usuário que está caminhando.

Os campos coletados deste plugin são representados na tabela [14](#)

Tabela 14 – Campos coletados pelo Google Activity Recognition

Campo	Tipo de campo	Descrição do campo
timestamp	Timestamp	Horário da coleta
activity_name	Texto	Nome da atividade detectada
activity_type	Inteiro	Constante da atividade detectada
confidence	Inteiro	Probabilidade de acerto do movimento em porcentagem (0%-100%)
activities	Texto	Arranjo JSON com todas as atividades em potencial e as suas probabilidades

6 Resultados

Este capítulo relata os testes deste trabalho. Apresenta os scripts e os bancos de dados utilizados para coleta de dados do Twitter e alguns aspectos dos dados coletados. Por fim, apresenta os dados coletados com as ferramentas para coleta de contextos e um teste de enriquecimento semântico com LOD.

6.1 Coleta e análise de dados do Twitter

A aquisição dos tweets foi realizada através do script *SeMovGet_Tweet*, desenvolvido em Java no laboratório LISA. Ele se conecta com o Twitter através da API Twitter4j¹. No processo de coleta de dados são executadas duas instâncias do script. Uma das instâncias executa a versão *Stream*, a qual realiza a coleta de tweets em tempo real. A outra instância executa a versão *Timeline* que realiza a coleta dos dados históricos de tweets dos usuários com maior número de tweets encontrados através de *Stream*.

As duas instâncias salvam os dados coletados em um banco de dados MongoDB por questão de eficiência. Como é mostrado em (JUNG; YOUN; BAE, 2015), o tempo da operação de inserção em bancos de dados PostgreSQL é muito maior do que em um banco de dados MongoDB, principalmente para grandes quantidades de registros.

Os dados coletados são posteriormente exportados para arquivos CSVs através do script *SeMovGet_Tweet_Backup*. Estes CSVs são utilizados para popular o servidor PostGIS que possui o esquema ilustrado na figura 4 utilizando o script *SeMovGet_Tweet_Restore*. Este procedimento é necessário para a realização de consultas com junções geoespaciais. O diagrama de atividades na figura 18 ilustra as etapas dos processos de coleta e carga dos dados.

¹ <http://twitter4j.org/>

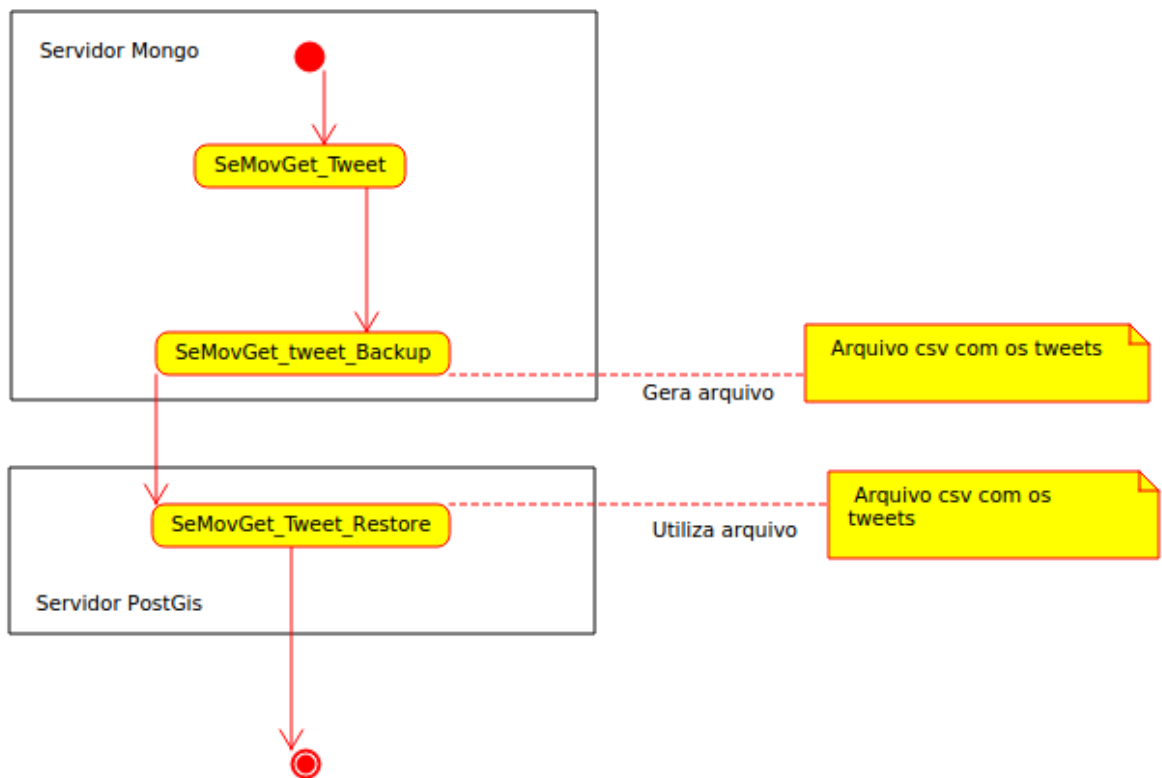


Figura 18 – Diagrama de atividades do processo de coleta e carga dos dados

6.1.1 Caracterização dos dados coletados

Dentre os dados coletados estão os dados de perfil dos usuários, dos tweets que este usuário postou e do *Foursquare* sobre a localização referenciada nas postagens. Em um total de 436.500.628 Tweets coletados, apenas 8,08% possuem coordenadas geográficas como ilustra o gráfico da imagem 19



Figura 19 – Gráfico comparando a quantidade de Tweets com e sem coordenadas geográficas

6.2 Testes com ferramentas para coleta de contextos

Para a realização dos testes foram coletados dados de usuários do Twitter e de objetos móveis com o Tweets Context Collector. Para a coleta dos dados do Twitter e junção dos dados do Twitter com os dados dos objetos móveis, foi criado um script em Python. Neste script são inseridos os nomes de usuários do Twitter dos participantes do testes e os IDs gerados pelo Framework AWARE de seus objetos móveis. O script se conecta à API do Twitter através da biblioteca Tweepy e coleta os dados do usuário e tweets de cada participante do teste. O Script então se conecta à uma base de dados MySQL que possui os dados coletados dos objetos móveis e então agrupa os dados dos sensores que estão relacionados ao tempo e ao objeto móvel de cada postagem. Por fim, os dados de usuário e Tweets que possuem dados de sensores relacionados são salvos em uma base de dados Postgres.

A base de dados que contém os dados do usuário, dos tweets e dos sensores é uma versão modificada da base de dados do capítulo 3. Nesta nova versão são adicionadas tabelas para cada sensor e essas tabelas são relacionadas com a tabela Post através do campo idPost, como mostra a figura 20.

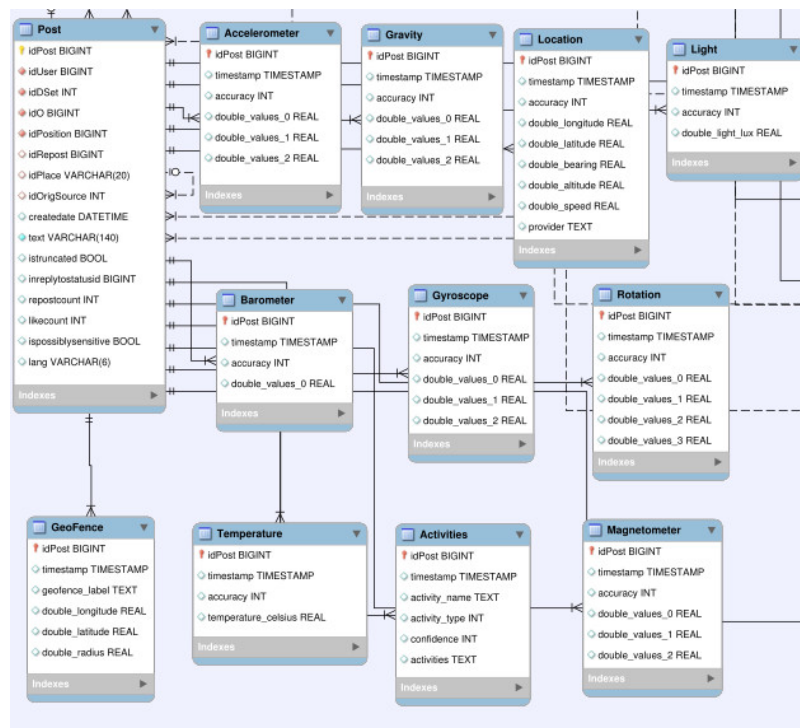


Figura 20 – Área modificada do modelo do banco de dados do capítulo 3

Para a realização de uma amostra, foi selecionado o tweet ilustrado na figura 21. Este tweet, como a maioria dos Tweets coletados, não possui a coordenada informando o local exato do objeto móvel, porém ele possui uma relação com o local Florianópolis. Esta

relação pode não ser realmente relacionada com a localização do usuário². Mas assumindo que o usuário marcou a localização em que estava, podemos utilizar os dados do polígono coletado do lugar Florianópolis pela API do Twitter e afirmar que o usuário está dentro da região retangular real marcada no mapa da figura 22.

Me arrumando pra ir pra Copel comprar um microondas para o novo apartamento

11:09 - 6 de mai de 2017 de Florianópolis, Brasil



Figura 21 – Tweet utilizado para exemplificar a utilidade do Tweet Context Collector

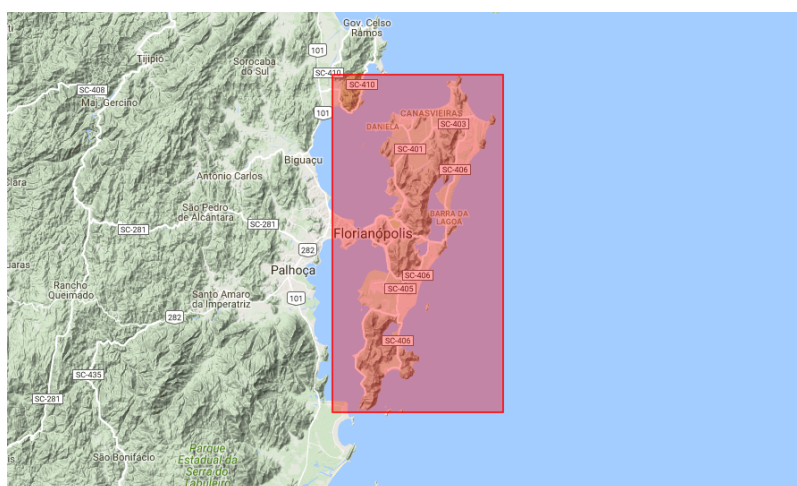


Figura 22 – Mapa gerado pela API do Google Maps com um polígono mostrando a área de Florianópolis definido pelo Twitter

Com o Tweets Context Collector, além dos dados coletados pela API do Twitter é possível especificar melhor a localização do usuário com a ajuda do plugin Google Fused Location. Utilizando o exemplo da figura 21, com o Tweet Context Collector foi possível coletar a longitude, latitude e a precisão dos dados, gerando um círculo que se aproxima muito mais da coordenada real do usuário que o retângulo envolvendo Florianópolis, com erro máximo de apenas 20 metros, como mostra na figura 23.

Além disso, graças ao plugin Google Activity Recognition, é possível identificar que o objeto móvel estava parado, com 100% de precisão. Com os dados coletados com os sensores magnetômetro e acelerômetro, é possível descobrir o azimute do objeto móvel para identificar a orientação do usuário, como mostra a figura 24.

² <https://dev.twitter.com/overview/api/places>

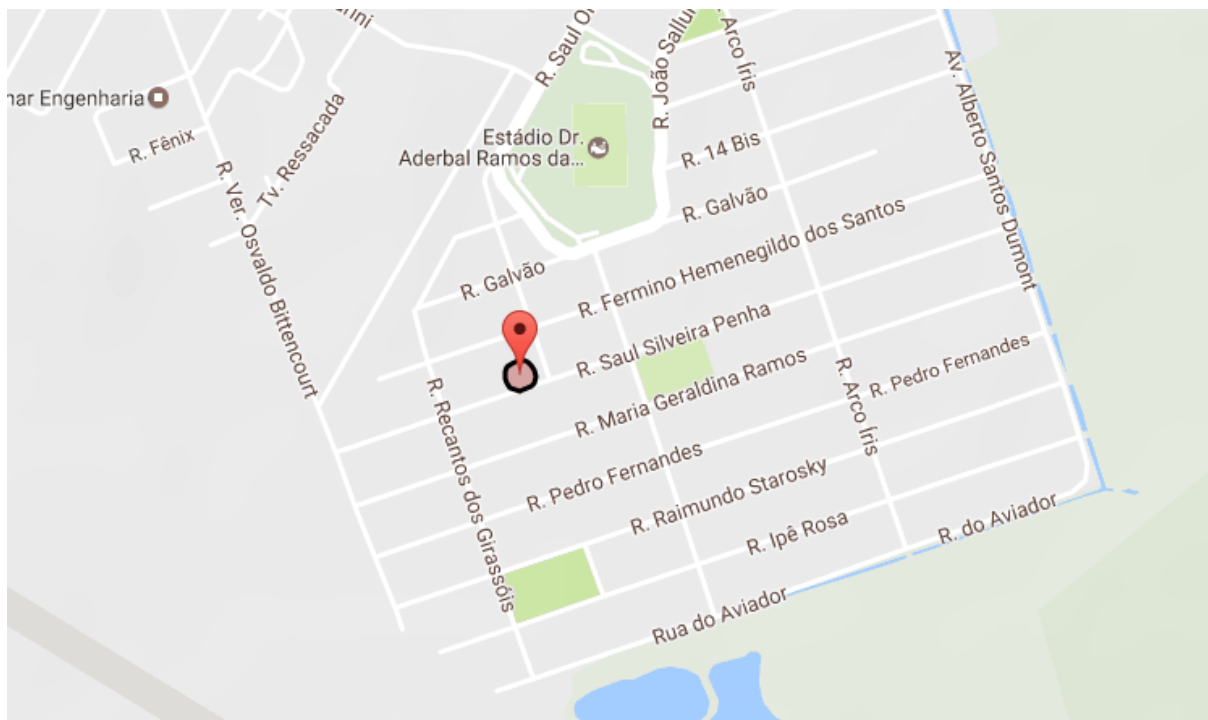


Figura 23 – Mapa gerado pela API do google maps com o raio aproximado da localização do objeto móvel.

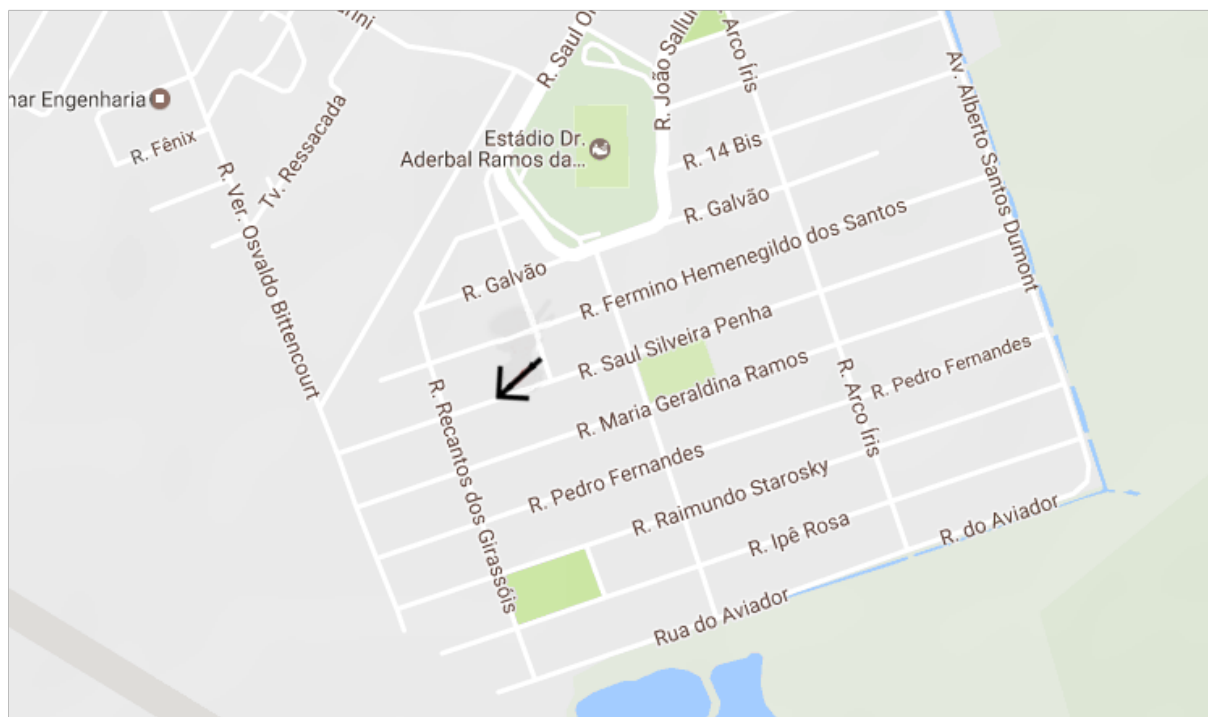


Figura 24 – Mapa gerado pela API do google maps com a orientação do usuário

6.2.1 Teste de enriquecimento semântico com LOD

Para o teste de enriquecimento semântico LOD foi utilizado o tweet da figura 25. Este tweet, tal como o exemplo da figura 22, possui apenas identificação de local, que pode

ser aproximado pelo polígono envolvendo Florianópolis. Também foi utilizado o algoritmo de ligação de postagens a locais proposto junto ao Baquara (FILETO et al., 2015).

Saindo da UFSC

18:16 - 30 de mai de 2017 de Florianópolis, Brasil



Figura 25 – Tweet utilizado para o teste

Utilizando o Tweet Context Collector foi possível identificar a localização do objeto móvel no momento da postagem. A localização está representada na figura 26. Também foi utilizado o recurso do dbpedia "Universidade Federal de Santa Catarina"³.

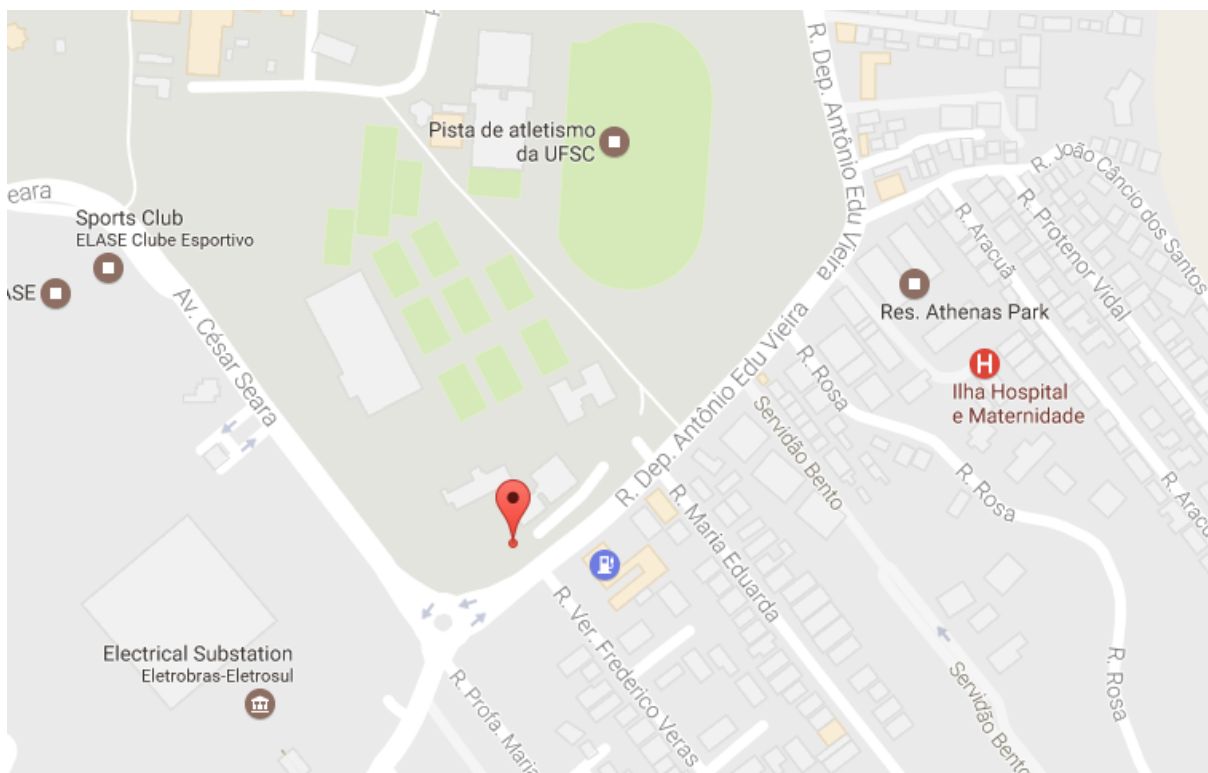


Figura 26 – Tweet utilizado para o teste

Como demonstrado na figura 27, o algoritmo do Baquara compara a distância entre o recurso e o objeto móvel e a similaridade do texto do tweet com o nome do recurso e exige valores mínimos e máximos para verificar se o recurso deve ser associado ao texto. Utilizamos uma distância máxima de 700m e uma similaridade mínima de 50%.

Como demonstrado na figura 28, a distância entre o objeto móvel e a UFSC é de 680,04 metros. Utilizando a métrica para comparação de strings Jaro-Winkler, descobri-

³ http://es.dbpedia.org/page/Universidad_Federal_de_Santa_Catarina

```

input :  $S = \{s_0, \dots, s_n\}$ ; // Pre-processed movement segments
          $R = \{r_0, \dots, r_m\}$ ; // Pre-processed resources set
          $\tau_s \in \mathbb{R}^+$ ; // Spatial distance threshold in meters
          $\tau_t \in \mathbb{R}^+$ ; // Textual similarity threshold

output:  $SA$ ; // Semantic annotations of movement segments in  $S$ 

1 begin
2    $SA \leftarrow \emptyset$ ; // Semantic Annotations ( $SA$ ) set initially empty
3    $SJ \leftarrow (\Pi_{s \leftarrow S, r \leftarrow R, geoDist(S \bowtie_{(geoDist \leftarrow dist(s.geom, r.geom)) \leq \tau_s} R))$ ;
4   foreach  $s \in S$  do
5      $k \leftarrow 0$ ; // Initialize best matching measures for  $s$ 
6      $minDist \leftarrow \tau_s$ ;
7      $maxSim \leftarrow \tau_t$ ;
8     foreach  $(s, r, geoDist) \in SJ$  do
9       if  $geoDist \leq minDist$  then
10         $textSim \leftarrow textualSimilarity(s.ppText, r.ppText)$ ;
11        if  $textSim \geq maxSim$  then
12          if  $geoDist < minDist \vee textSim > maxSim$  then
13             $k \leftarrow 0$ ; // Better matching resource  $r$  found
14             $minDist \leftarrow geoDist$ ;
15             $maxSim \leftarrow textSim$ ;
16             $k++$ ; // Increment number of matchings
17             $bestMatching[k] \leftarrow r$ ; // Add matching  $r$ 
18        while  $k > 0$  do
19           $k--$ ; // Create semantic annotations for segment  $s$ 
20           $SA \leftarrow SA \cup (s, visits, bestMatching[k])$ ;
21 return  $SA$ ;

```

Figura 27 – Algoritmo proposto em (FILETO et al., 2015)

mos que a similaridade entre UFSC e o nome do recurso "Universidade Federal de Santa Catarina" é de 51,75%. Como tanto a distância é menor que a máxima, quanto a similaridade é maior que a mínima, o recurso serve para o enriquecimento semântico do tweet. No entanto, obviamente, um simples dicionário de nomes de superfície correlacionado tais strings resolveria o problema de casamento entre o nome por extenso e o acrônimo da UFSC.

Além disso, como o objeto móvel estava em movimento, o sensor de localização detectou que a velocidade era de 1.5 metros por segundo (5.4 km/h) e que a orientação era de 224.5 graus, como representado na figura 29. Os dados retirados a partir do plugin Google Activity Recognition também acrescentam dados, indicando que o usuário estava se movimentando a pé.

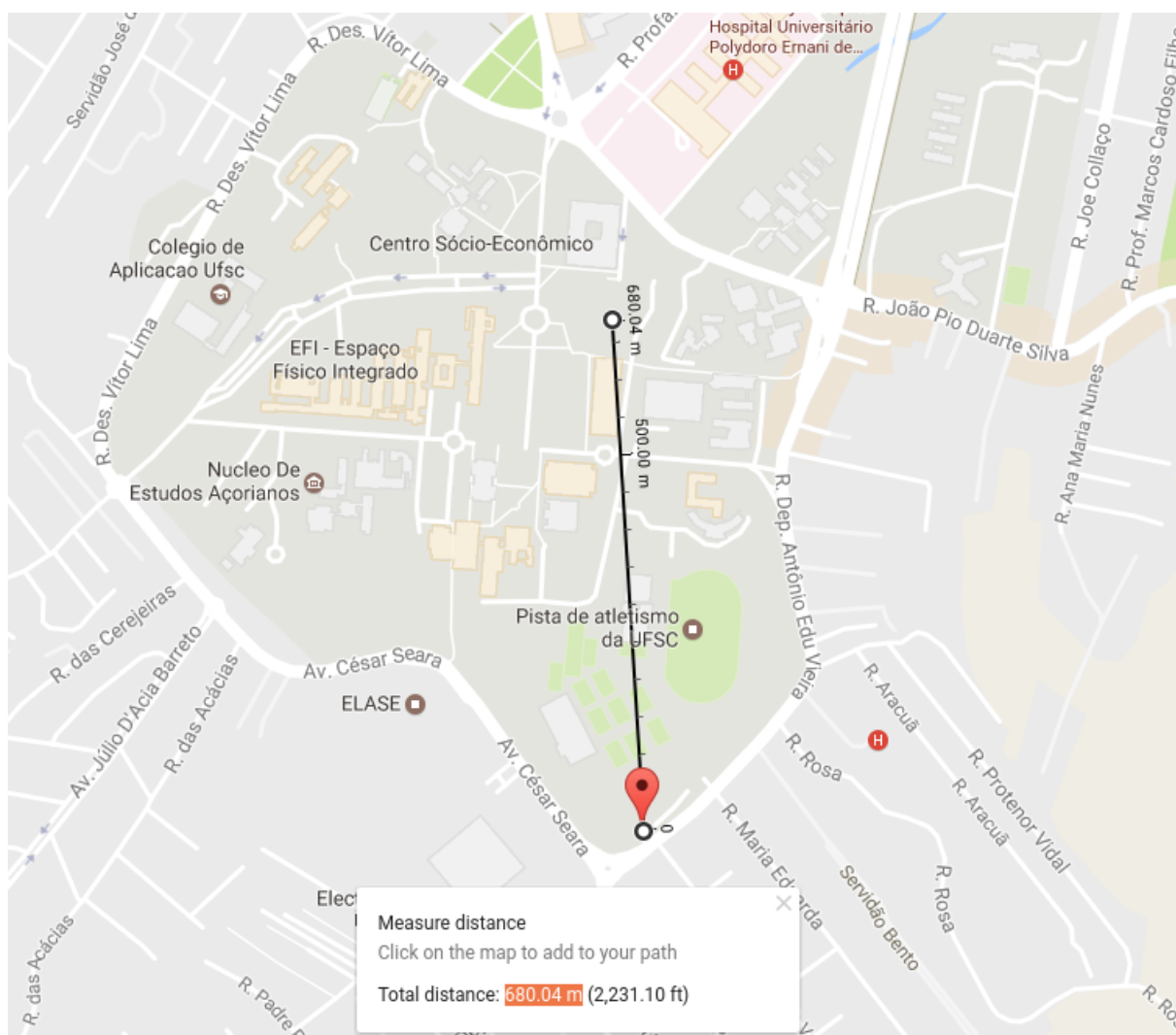


Figura 28 – Distância calculada pela API do Google Maps

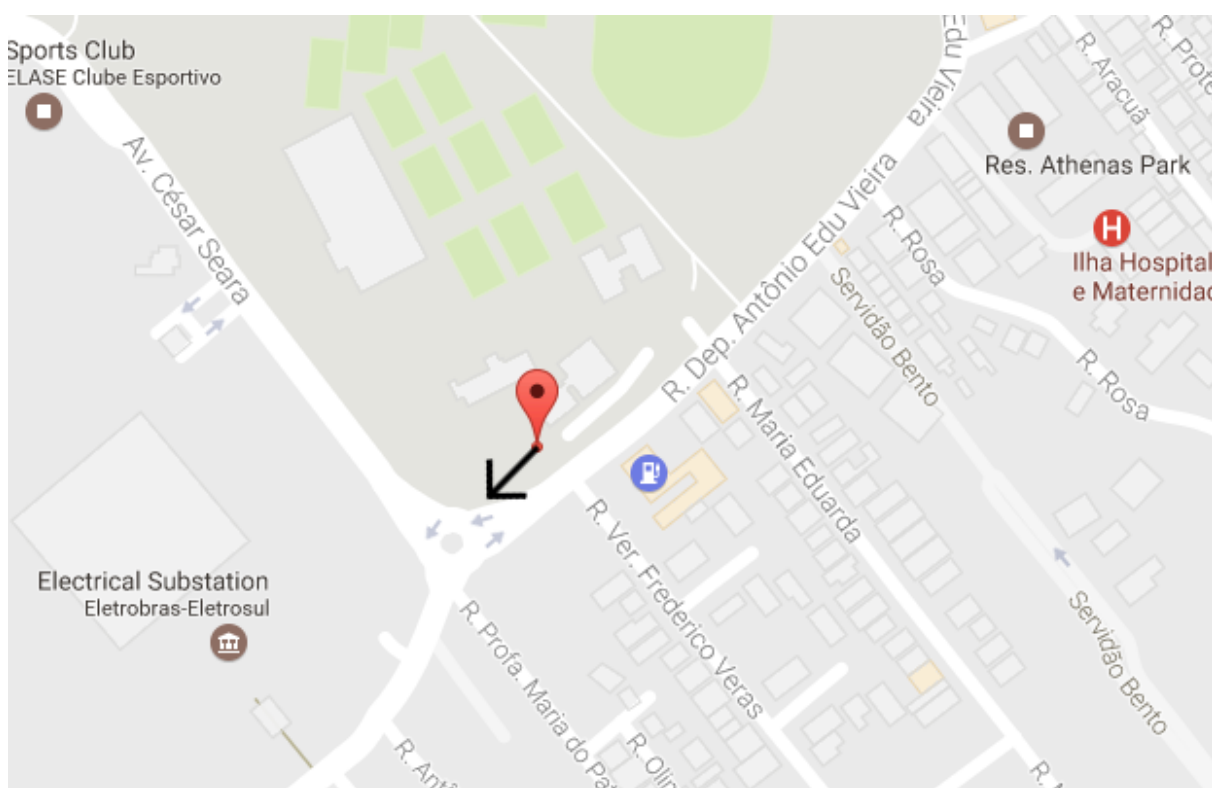


Figura 29 – Coordenada e orientação do usuário no momento da postagem do Tweet

7 Conclusões e trabalhos futuros

Este trabalho apresentou uma revisão bibliográfica com os conceitos necessários ao entendimento de questões referentes ao enriquecimento semântico de dados sobre objetos móveis e o uso de informação de contexto em tal enriquecimento. Ele propôs melhorias em um esquema de bancos de dados para experimentos de enriquecimento semântico. Anotações semânticas, que são a base de tal enriquecimento, podem ser feitas e/ou sua qualidade aferida mediante o uso de dados sobre o contexto desses objetos móveis. Assim, este trabalho também apresentou um estudo do estado da arte sobre ferramentas de coleta de informação sobre o contexto de objetos móveis. Finalmente, apresentou um aplicativo desenvolvido para complementação de dados coletados de objetos móveis com informação de contexto e reportou alguns testes realizados para validar tal aplicativo. Assim, as contribuições deste trabalho são:

1. revisão e melhoramentos em um esquema de banco de dados para acomodar dados e resultados de experimentos de anotação semântica;
2. revisão de um coletor de dados de tweets visando acomodação dos dados colhidos em um banco de dados com tal esquema;
3. estudo comparativo de algumas das principais ferramentas para coleta de informação sobre o contexto de objetos móveis;
4. desenvolvimento de um aplicativo baseado em uma ferramenta de coleta de dados sobre o contexto para complementar dados coletados de mídias sociais com dados como coordenadas geográficas;
5. Um teste caracterizando dados coletados do Twitter e o uso do aplicativo para associar coordenadas geográficas a tweets que não as incluíam.

O teste mostrou que a maioria dos tweets coletados não possui coordenadas geográficas, que são necessárias a alguns processos de enriquecimento semântico. O aplicativo baseado em coleta de dados de contexto foi capaz de obter essas coordenadas e associar aos tweets que não as tinha para então realizar o processo de anotação semântica.

Para trabalhos futuros sugere-se:

1. pesquisar formas de aprimorar o Tweet Context Collector, diminuindo o consumo de bateria e conectando o com outros plugins;
2. realizar experimentos com mais voluntários e maiores amostras de tweets;

3. criar regras ouro para o enriquecimento semântico de tweets.

Referências

- ACHREKAR, H. et al. Predicting flu trends using twitter data. *The First International Workshop on Cyber-Physical Networking Systems*, 2011. Citado na página 13.
- AGARWAL, A. et al. Sentiment analysis of twitter data. *LSM '11 Proceedings of the Workshop on Languages in Social Media*, 2011. Citado na página 13.
- BOLCHINI, C. et al. And what can context do for data? *Communications of the ACM - Scratch Programming for All CACM*, 2009. Citado na página 17.
- BRUNETTE, W. et al. Open data kit 2.0: Expanding and refining information services for developing regions. *HotMobile '13 Proceedings of the 14th Workshop on Mobile Computing Systems and Applications Article No. 10*, 2013. Citado na página 28.
- CHEN, G.; KOTZ, D. And what can context do for data? *A Survey of Context-Aware Mobile Computing Research*, 2000. Citado na página 17.
- CHEN, Z. et al. gmission: A general spatial crowdsourcing platform. *ICTD '10 Proceedings of the 4th ACM/IEEE International Conference on Information and Communication Technologies and Development*, 2010. Citado na página 32.
- CHITTILAPPILLY, A. I.; CHEN, L.; AMER-YAHIA, S. A survey of general-purpose crowdsourcing techniques. *IEEE Transactions on Knowledge and Data Engineering (Volume: 28, Issue: 9, Sept. 2016)*, 2016. Citado na página 32.
- DEY, A. K.; ABOWD, G. D. Towards a better understanding of context and context-awareness. *HUC '99 Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, 2000. Citado na página 17.
- DUARTE, P. A. de S. et al. A model-driven approach to generate context aware applications. *WebMedia '14 Proceedings of the 20th Brazilian Symposium on Multimedia and the Web Pages 99-102*, 2014. Citado na página 33.
- DUARTE, P. A. S. et al. Dynamic deployment for context-aware multimedia environments. *WebMedia '15 Proceedings of the 21st Brazilian Symposium on Multimedia and the Web Pages 197-204*, 2015. Citado na página 33.
- FERREIRA, D. *AWARE: A mobile context instrumentation middleware to collaboratively understand human behavior*. Dissertação (Mestrado) — University of Oulu, 2013. Citado na página 34.
- FERREIRA, D.; KOSTAKOS, V.; DEY, A. K. Aware: mobile context instrumentation framework. *WebMedia '14 Proceedings of the 20th Brazilian Symposium on Multimedia and the Web Pages 99-102*, 2014. Citado na página 34.
- FILETO, R. et al. The baquara2 knowledge-based framework for semantic enrichment and analysis of movement data. *Data & Knowledge Engineering*, 2015. Citado 4 vezes nas páginas 7, 17, 52 e 53.

- FRUTUOSO, D. G. *Recuperação de informação e classificação de entidades organizacionais em textos não estruturados*. Dissertação (Mestrado) — UFPE, 2014. Citado na página 13.
- GARRITTY, C.; EMAM, K. E. Who's using pdas? estimates of pda use by health care providers: a systematic review of surveys. *J Med Internet Res*, 2006. Citado na página 13.
- HARTUNG, C. et al. Open data kit: Tools to build information services for developing regions. *ICTD '10 Proceedings of the 4th ACM/IEEE International Conference on Information and Communication Technologies and Development*, 2010. Citado 3 vezes nas páginas 7, 25 e 27.
- JUNG, M.-G.; YOUN, S.-A.; BAE, J. A study on data input and output performance comparison of mongodb and postgresql in the big data environment. *2015 8th International Conference on Database Theory and Application (DTA)*, 2015. Citado na página 47.
- KOULOUMPIS, E.; WILSON, T.; MOORE, J. Twitter sentiment analysis: the good the bad and the omg! *Fifth International AAAI Conference on Weblogs and Social Media*, 2011. Citado na página 13.
- KUMAR, S.; MORSTATTER, F.; LIU, H. *Twitter Data Analytics*. [S.l.]: Springer, 2014. 1-69 p. (Springer Briefs in Computer Science). ISBN 978-1-4614-9371-6. Citado na página 13.
- LEHMANN, J. et al. Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, v. 6, n. 2, p. 167–195, 2015. Disponível em: <<http://dx.doi.org/10.3233/SW-140134>>. Citado na página 20.
- MATHIOUDAKIS, M.; KOUDAS, N. Twittermonitor: Trend detection over the twitter stream. *International Conference on Management of data*, 2010. Citado na página 13.
- NABO, R. G. B. et al. Anotação de trajetórias via fusão com trilhas de mídias sociais. *GeoInfo*, 2014. Citado na página 16.
- NGOMO, A. N. et al. Introduction to linked data and its lifecycle on the web. In: KOUBARAKIS, M. et al. (Ed.). *Reasoning on the Web in the Big Data Era, Athens, Greece*. Springer, 2014. (Lecture Notes in Computer Science, v. 8714), p. 1–99. ISBN 978-3-319-10586-4. Disponível em: <http://dx.doi.org/10.1007/978-3-319-10587-1_1>. Citado na página 19.
- OLIVEIRA, O.; BOLLIGER, F.; FLORIDO, A. Brazil agricultural census 2006: Innovations and impacts. Fourth International Conference on Agriculture Statistics, 2006. Citado na página 13.
- PAUL, M. J.; DREDZE, M. You are what you tweet: Analyzing twitter for public health. *Fifth International AAAI Conference on Weblogs and Social Media*, 2011. Citado na página 13.
- SCHILIT, B.; ADAMS, N.; WANT, R. Context-aware computing applications. *1st International Workshop on Mobile Computing Systems and Applications*, 1994. Citado na página 17.

STADLER, C. et al. Linkedgeodata: A core for a web of spatial open data. *Semantic Web*, v. 3, n. 4, p. 333–354, 2012. Disponível em: <<http://dx.doi.org/10.3233/SW-2011-0052>>. Citado na página 20.

SUNDARAM, H.; MANI, A. Context-aware dynamic presentation synthesis for exploratory multimodal environments. *2005 IEEE International Conference on Multimedia and Expo*, 2005. Citado na página 17.

WU, F. et al. Semantic annotation of mobility data using social media. *24th International Conference on World Wide Web*, 2015. Citado na página 14.

APÊNDICE A – Artigo sobre o TCC

Análise, seleção e teste de ferramentas para coleta de dados sobre objetos móveis visando enriquecimento semântico

Karran Besen¹, Renato Fileto¹

¹Curso de bacharelado em Ciências da Computação – Universidade Federal de Santa Catarina (UFSC) – 88.040-900 – Florianópolis – SC – Brazil

{karran@inf., r.fileto@}ufsc.br

Abstract. *Many of the data generated by the use of social media can feed a wide variety of applications. However, such postings have structured and non-structured content, subject to noise and interpretation problems. So, these posts need to be semantically enriched before being used in certain applications. To generate quality annotations, semantic enrichment of these posts requires information about the context within which such posts are made. This article presents a review of the literature on tools for collecting additional contextual information that may assist in semantic enrichment and an application proposal for collecting data on mobile objects at the time of Twitter postings*

Resumo. *Muitos dos dados gerados pelo uso de mídias sociais podem alimentar uma ampla variedade de aplicações. Entretanto, tais postagens têm conteúdo textual não estruturado, sujeito a ruídos e problemas de interpretação. Assim, estas postagens precisam ser semanticamente enriquecidas antes de serem utilizadas em certas aplicações. Para gerar anotações de qualidade, o enriquecimento semântico destas postagens requer informações sobre o contexto dentro do qual tais postagens são feitas. Este artigo apresenta uma revisão de literatura sobre ferramentas para coleta de informações adicionais de contexto que possam auxiliar no enriquecimento semântico e uma proposta de aplicativo para coleta de dados sobre objetos móveis no momento de postagens no Twitter.*

1. Introdução

O uso de objetos móveis para a coleta de dados não é novidade, notebooks e palm-tops já são usados a muito tempo para coleta de dados, como por exemplo na saúde [Garrity and Emam 2006], na agricultura [Oliveira et al. 2006], etc. A redução do custo de aquisição e operação e aumento de funcionalidades de smartphones tornou estes objetos móveis disponíveis até em regiões extremamente pobres e remotas, motiva ainda mais a exploração do uso destes dispositivos na coleta de dados.

Até os celulares mais simples podem ser utilizados na coleta de dados, por exemplo com o uso de serviços de mensagem de texto (e.g. SMS) ou voz. Os smartphones podem servir de formas muito mais sofisticadas para estas coletas, como por exemplo apresentando formulários que possuem interfaces amigáveis, utilizando os dados gerados pelos sensores do dispositivo ou então os dados gerados pela interação com mídias sociais.

A quantidade de conteúdo gerado pelas mídias sociais (e.g. Twitter, Facebook, Instagram, Foursquare) aumenta constantemente. Nas mídias sociais os usuários se tornam fontes de dados. O Twitter, por exemplo, possui atualmente cerca de 310 milhões

de usuários ativos¹, i.e., que enviam tweets mensalmente. Esses usuários muitas vezes relatam informações e expressam suas opiniões e sentimentos através de tais tweets, que ficam em grande parte disponíveis via diversas interfaces, incluindo APIs² e frameworks de coleta de dados [Kumar et al. 2014]. Assim, mídias sociais têm se tornado imensas e ricas fontes de informação para diversas aplicações.

Os dados das mídias sociais são utilizados em uma grande diversidade de pesquisas, tais como recuperação e análise de informação [Frutuoso 2014], análise de sentimentos [Kouloumpis et al. 2011, Agarwal et al. 2011] e recomendação [Mathioudakis and Koudas 2010]. As informações adquiridas das mídias sociais podem ser úteis em vários domínios de aplicação, tais como marketing, turismo e segurança pública [Achrekar et al. 2011, Paul and Dredze 2011].

Entretanto, dados de mídias sociais apresentam uma enorme variedade de formatos, podendo incluir dados semiestruturados (e.g., tags, timestamps, coordenadas geográficas) e não estruturados (textos livres). Os textos postados em mídias sociais normalmente são informais, sujeitos a muitos erros ortográficos e gramaticais, gírias e acrônimos (e.g. LOL - Lots of laughs, sqn - Só que não, kd - cadê, vc - você), entre outros problemas.

Devido à estrutura linguística irregular, postagens podem ser interpretadas de forma errônea ou com pouca precisão. Assim, para se extrair informação de qualidade, torna-se necessário enriquecer as postagens com referências semânticas bem definidas antes de utilizá-las. A combinação de dados de contexto dos objetos móveis (e.g., informações de perfil) e com os contextos e conteúdos das postagens em mídias sociais tem potencial para produzir anotações semânticas de qualidade, de modo a permitir a interpretação correta dos dados. Com estas anotações semânticas pode-se, por exemplo, desambiguar e identificar recursos para investigar melhor o motivo da viagem de um grupo de pessoas para uma localização em um determinado horário (e.g., Ir a um festival de música, assistir a uma partida de futebol)[Wu et al. 2015].

Tendo em vista que o processo de enriquecimento semântico é essencial para a aplicação dos dados coletados de mídias sociais e que informações de contexto podem auxiliar neste processo, este trabalho propõe a revisão da literatura sobre ferramentas para coleta de informações adicionais de contexto que possam aprimorar o enriquecimento semântico.

2. Ferramentas para coleta de contextos

Esta seção descreve ferramentas, incluindo bibliotecas, suítes e frameworks, para a coleta de informações de contexto de objetos móveis, de acordo com a documentação encontrada na literatura.

Tais ferramentas são apresentadas e comparadas de acordo com os sistemas operacionais que conseguem executar a ferramenta, com as definições de autorização de uso da ferramenta, com as formas em que os dados podem ser coletados pela ferramenta, com os formatos em que estes dados coletados podem ser recuperados, com as ações que o programa consegue desempenhar (e.g. Criação de formulários, mapeamento de dados coletados, etc.) e pelas limitações e os requisitos que a ferramenta exige para ser executada.

¹<https://about.twitter.com/company>

²<https://dev.twitter.com/overview/documentation>

2.1. Open Data Kit (ODK)

O ODK³ é uma suíte de ferramentas com o intuito de coletar e gerenciar dados de objetos móveis. Foi desenvolvido pela Universidade de Maryland e pelo Instituto Internacional de análise de sistemas aplicados. A suíte permite a criação de formulários com a ferramenta ODK Build, a hospedagem destes formulários em um servidor com o ODK Aggregate e o preenchimento e envio destes formulários para o servidor através de smartphones com o ODK Collect.

De acordo com [Hartung et al. 2010], foi escolhido o padrão *xForms*⁴, para que todas as ferramentas possam ser usadas independentemente, mas podendo interagir entre elas. O *xForms* é um padrão de descrições de formulários desenvolvido pela *World Wide Web consortium (W3C)*⁵. Esta padronização permite que o ODK importe os formulários de outras ferramentas, como por exemplo o *XLSForm*⁶ e o *koboform*⁷.

2.2. GeoODK

A GeoODK é uma modificação Opensource do ODK feita pelo Instituto Internacional de Análise de Sistemas Aplicados e pela Global Agricultural Monitoring. A GeoODK fornece uma maneira de coletar e armazenar informações geo-referenciadas. Esta suíte também fornece ferramentas para visualização, análise e manipulação dos dados.

Os grandes diferenciais do GeoODK são as funcionalidades de mapeamento existentes na GeoODK. Entre as funcionalidades de mapeamento adicionais estão o mapeamento offline a possibilidade de visualização dos dados coletados no dispositivo móvel e a coleta de polígonos e traços do GPS.

Além disto, a GeoODK fornece o Mobile Data Converter, que é um software que permite a conversão de dados coletados para formatos espaciais e a preparar um conjunto de dados para um sistema GIS.

2.3. KoBoToolbox

A KoBoToolbox é uma suíte de ferramentas Opensource baseadas no ODK para coleta de dados de objetos móveis para organizações humanitárias e pesquisadores. Foi desenvolvido pela Harvard Humanitarian Initiative. A suíte é composta pelas ferramentas:

- koboform (dkobo, django kobo) - Ferramenta web para criar os formulários;
- kobocat and kobocat-templates - Servidor para hospedar formulários e analisar os dados;
- enketo-express - Aplicação Web, desenvolvida em HTML para coletar, pré-visualizar formulários e editar os dados submetidos;
- kobocollect - Aplicativo android para coleta de dados.

2.4. gMission

O aplicativo gMission foi desenvolvido pela Hong Kong University of Science and Technology e tem como objetivo crowdsourcing espacial. O gMission é opensource e está

³<https://opendatakit.org/>

⁴<http://w3.org/TR/xforms/>

⁵<https://www.w3.org>

⁶<http://xlsform.org/>

⁷<http://kobotoolbox.org/>

disponível para Android e iOS. Existe pouca documentação e usuários que utilizam o gMission.

O gMission conta com um sistema de contribuição explicado em [Chittilappilly et al. 2016]. Nele para quanto mais tarefas forem cumpridas, mais pontos são adquiridos. Quanto mais pontos uma pessoa tiver, mais perguntas podem ser criadas.

De acordo [Chen et al. 2010], o gMission possui a interface de usuário que é responsável por fornecer ao usuário as funções de postar e responder tarefas, e o gerenciador de funções que é responsável pela localização do usuário, pela recomendação das tarefas a partir de informações geográficas e da quantidade de tarefas e também pelo controle de qualidade.

2.5. LoCCAM

O LoCCAM⁸ é um middleware para sistemas móveis e sensíveis ao contexto desenvolvido pelo Grupo de Redes de Computadores e pela Engenharia de Software e Sistemas (GREat) da Universidade Federal do Ceará (UFC)[Duarte et al. 2015]. Com o LoCCAM é possível desenvolver aplicações sensíveis ao contexto rapidamente, como por exemplo uma aplicação com o objetivo de mapear determinados locais com informações referentes à localização do dispositivo⁹. A arquitetura do LoCCAM, ilustrada na figura 1 retirada do site do LoCCAM¹⁰, pode ser dividida em duas partes principais [de Sousa Duarte et al. 2014]:

- **SysSU (System Support for Ubiquity):** Recebe a relação de interesses das aplicações e as repassa para o CAM;
- **CAM (Context Acquisition Manager) Framework :** armazena a relação de interesses e controla os CACs (Componentes de aquisição de contexto).

O SysSU Filter realiza o filtro nas informações dos sensores, o adaptation reasoner cuida da lista de interesses da aplicação e se comunica com o CAC manager que controla todos os CACs.

⁸<http://loccam.great.ufc.br/>

⁹<http://loccam.great.ufc.br/downloads/download-aplicacoes.html>

¹⁰<http://loccam.great.ufc.br/informacoes-gerais/arquitetura.html>

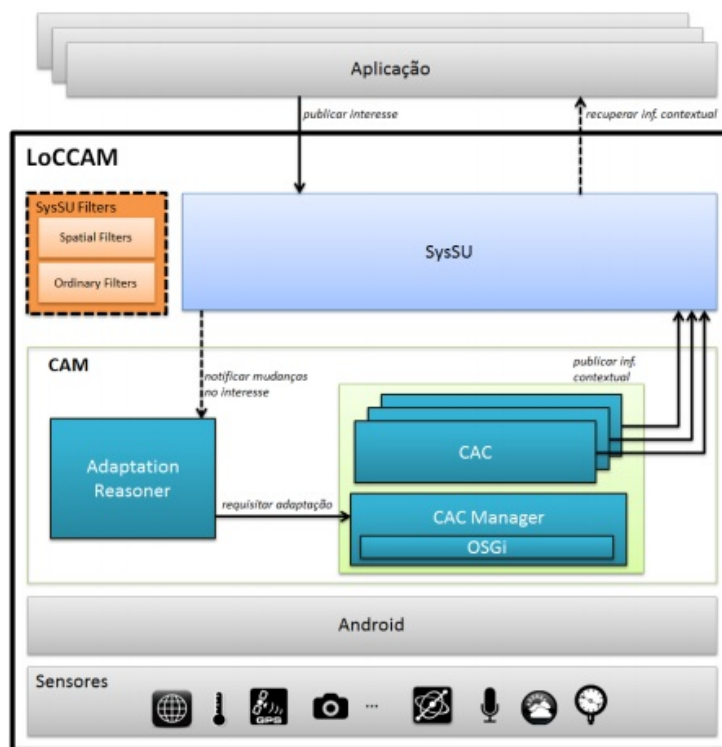


Figura 1. Arquitetura do LoCCAM

2.6. AWARE

O AWARE¹¹ é um middleware open-source dedicado a captura de informações de contexto móvel desenvolvido na Carnegie Mellon University. Pode ser utilizado para criar aplicativos para sistemas operacionais Android e iOS. Com ele é possível criar aplicações sensíveis ao contexto e coletar dados de contexto.

A figura 2 (retirada de [Ferreira et al. 2014]) ilustra a arquitetura do AWARE. O AWARE utiliza **sensores** como forma de capturar dados de contextos de hardware (e.g., Acelerômetro, giroscópio, magnetismo), software (e.g., calendário, email, status das aplicações) e humanos (e.g., ESM - Experience sampling Method, entradas de voz e gesto). O AWARE possui **plugins**, que são extensões que permitem os desenvolvedores implementar códigos que utilizam sensores ou até mesmo outros plugins. Um plugin deve refinar o contexto oferecido por algum sensor, fornecendo contextos mais precisos ou mais compreensíveis. Para a comunicação entre sensores, plugins e aplicações são utilizadas as estratégias de **broadcast**, **observador** e **fornecedor**.

Por fim, o dispositivo pode enviar os dados adquiridos dos sensores para outros dispositivos por MQTT (Padrão de telemetria) ou para um banco de dados MySQL por HTTPS. Os dados podem ser armazenados gratuitamente no **AWARE Server**.

¹¹<http://www.awareframework.com/>

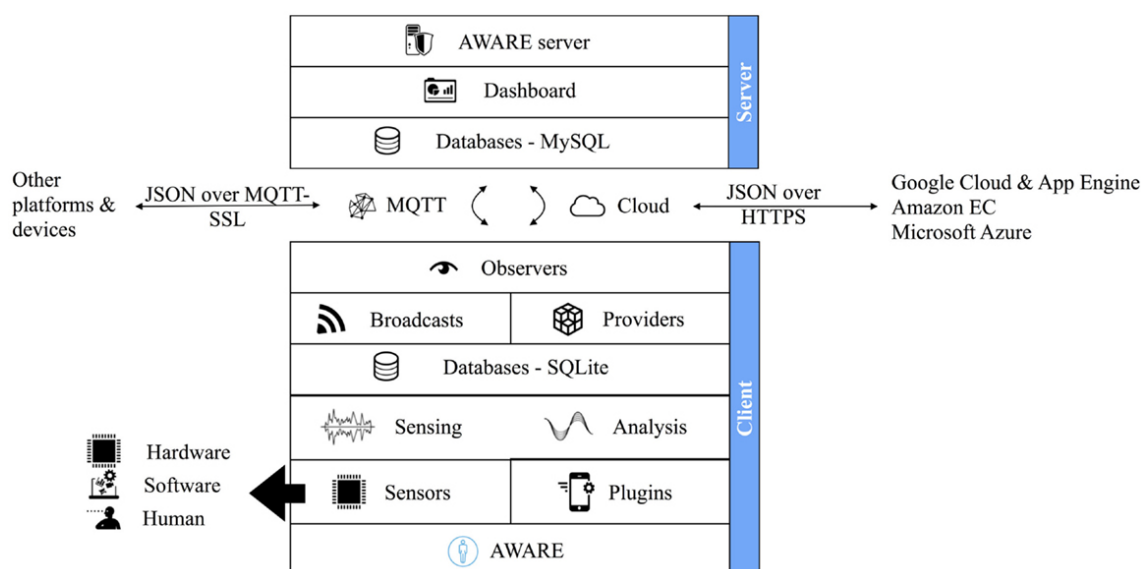


Figura 2. Arquitetura do AWARE

3. Síntese Comparativa

Nesta seção foram utilizados os critérios do início deste capítulo para realizar uma síntese comparativa das ferramentas estudadas. O objetivo da tabela 1 é comparar os tipos de licença e as plataformas, entradas e saídas suportadas pelas ferramentas estudadas.

Tabela 1. Comparação de ferramentas para coleta de dados sobre objetos móveis

Ferramentas	Plataformas	Licenças	Entradas	Saídas
ODK	Android	Opensource	Formulários	KML, JSON, CSV
GeoODK	Android	Opensource	Formulários	ESRI, KML, JSON, CSV
KoboToolbox	Android, iOS	Opensource	Formulários	KML, JSON, CSV, SPSS
gMission	Android, iOS	Opensource	Formulários	-
LoCCAM	Android	Freeware	Sensores	Tuplas
AWARE	Android, iOS	Opensource	Formulários, Sensores	CSV

Na tabela 2 foram comparadas os campos e os sensores de entradas presentes nas ferramentas estudadas. As ferramentas ODK, GeoODK e KoBoToolbox aceitam os mesmos campos e por isso estão listadas na mesma coluna da tabela. Com as tabelas comparativas é possível notar que das ferramentas estudada que apenas suportam formulários

Tabela 2. Comparação dos campos e sensores das ferramentas estudadas

Formato de entrada	Campos	ODK, GeoODK, KoboToolbox	LoCCAM	AWARE
Sensores	Acelerômetro		X	X
	Aplicações			X
	Barômetro			X
	Comunicação			X
	Gravidade		X	X
	Giroscópio			X
	Instalações			X
	Luminosidade		X	X
	Aceleração linear			X
	Localização	X	X	X
	Magnetômetro		X	X
	Orientação			X
	Proximidade			X
	Rotação			X
	Telefonia			X
	Temperatura		X	X
Tráfego			X	
Wi-Fi			X	X
Formulário	Alternativa ou múltipla escolha	X		X
	Decimal	X		
	Inteiro	X		X
	Imagens	X		
	Códigos de barra	X		
	Áudio	X		
	Vídeo	X		
	Texto	X		X

como forma de entrada, apenas a KoboToolbox tem suporte aos sistemas operacionais Android e iOS. Também é possível notar que a única ferramenta estudada não opensource é o middleware LoCCAM.

Ainda com o uso das tabelas é possível identificar o AWARE como a ferramenta mais completa. O AWARE é suportado pelos sistemas operacionais Android e iOS, é a única das ferramentas estudadas com a capacidade da coleta de dados de sensores e formulários, e também é a ferramenta que possui a maior quantidade de sensores já implementados em seu núcleo.

4. Aplicativo desenvolvido: Tweet Context Collector

Por ser a ferramenta estudada mais completa, o AWARE foi utilizado para o desenvolvimento de um aplicativo para realização de testes de associações de coordenadas geográficas a postagens que não às possuíam . O aplicativo pode ser executado em sistemas

Android e para poder coletar e enviar as informações dos sensores, o aplicativo deve estar pelo menos rodando em segundo plano. Também é necessária a ativação do serviço de acessibilidade do AWARE framework.

Como o Twitter lança uma notificação no celular quando o usuário posta um Tweet, foi utilizado o sensor de aplicativos existente no núcleo do AWARE para criar um observador de notificações. Desta forma, o Tweet Context Collector identifica quando o Twitter cria uma notificação, e então ativa os sensores e envia os dados recentemente coletados para um banco de dados MySQL.

O aplicativo possui uma única tela, aonde é possível visualizar o identificador único do objeto móvel gerado pelo AWARE. Também é possível visualizar um botão que serve como alternativa para a ativação da função de coleta e envio dos dados dos sensores. Este botão foi adicionado pois em alguns casos os usuários esqueceram de executar o aplicativo e só lembraram depois de postar o Tweet.

Enquanto o aplicativo está sendo executado, os sensores de baixa frequência ficam ativos. Os sensores de baixa frequência ficam periodicamente gravando localmente os valores mais recentes. Para diminuir o consumo de bateria e a quantidade de registros, os sensores de alta frequência ficam inativos até que a função de coleta dos dados é executada. Quando a função de coleta de dados é executada, cada sensor de alta frequência é ativo por um segundo e os valores coletados são enviados para o servidor. Assim que todos os dados dos sensores são enviados para o servidor, os dados são apagados dos objetos móveis.

Em geral, os smartphones possuem diferentes sensores disponíveis. Por exemplo o sensor de temperatura é muito raro de ser encontrado nos celulares, porém ele está implementado e caso o aplicativo seja instalado em um celular com o sensor disponível os dados da temperatura serão coletados.

Os sensores implementados no Tweet Context Collector são:

- Acelerômetro;
- Barômetro;
- Gravidade;
- Giroscópio;
- Localização;
- Luminosidade;
- Magnetômetro;
- Rotação;
- Temperatura.

O aplicativo também foi integrado aos plugins de baixa frequência Google Activity Recognition e Google Fused Locations. O plugin Google Fused Location usa a API do Google Fused Location para fornecer a localização do usuário em uma forma eficiente energeticamente. O plugin Google Activity Recognition detecta o meio de transporte do usuário.

5. Resultados

Para a realização dos testes foram coletados dados de usuários do Twitter e de objetos móveis com o Tweets Context Collector. Para a coleta dos dados do Twitter e junção

dos dados do Twitter com os dados dos objetos móveis, foi criado um script em Python. Neste script são inseridos os nomes de usuários do Twitter dos participantes dos testes e os IDs gerados pelo Framework AWARE de seus objetos móveis. O script se conecta à API do Twitter através da biblioteca Tweepy e coleta os dados do usuário e tweets de cada participante do teste. O Script então se conecta à uma base de dados MySQL que possui os dados coletados dos objetos móveis e então agrupa os dados dos sensores que estão relacionados ao tempo e ao objeto móvel de cada postagem. Por fim, os dados de usuário e Tweets que possuem dados de sensores relacionados são salvos em uma base de dados Postgres.

Para a realização de uma amostra, foi selecionado o tweet ilustrado na figura 3. Este tweet, como a maioria dos Tweets coletados, não possui a coordenada informando o local exato do objeto móvel, porém ele possui uma relação com o local Florianópolis. Esta relação pode não ser realmente relacionada com a localização do usuário¹². Mas assumindo que o usuário marcou a localização em que estava, podemos utilizar os dados do polígono coletado do lugar Florianópolis pela API do Twitter e afirmar que o usuário está dentro da região retangular real marcada no mapa da figura 4.

Me arrumando pra ir pra Copel comprar um
microondas para o novo apartamento

11:09 - 6 de mai de 2017 de Florianópolis, Brasil



Figura 3. Tweet utilizado para exemplificar a utilidade do Tweet Context Collector

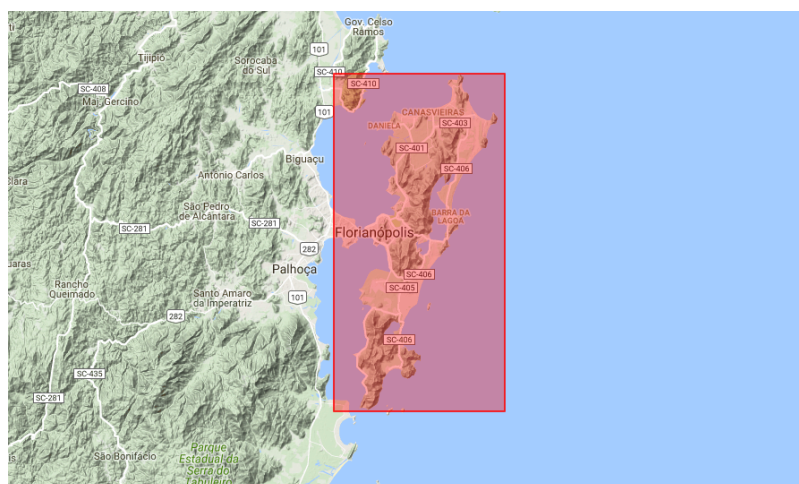


Figura 4. Mapa gerado pela API do Google Maps com um polígono mostrando a área de Florianópolis definido pelo Twitter

Com o Tweets Context Collector, além dos dados coletados pela API do Twitter é possível especificar melhor a localização do usuário com a ajuda do plugin Google Fused Location. Utilizando o exemplo da figura 3, com o Tweet Context Collector foi possível

¹²<https://dev.twitter.com/overview/api/places>

coletar a longitude, latitude e a precisão dos dados, gerando um círculo que se aproxima muito mais da coordenada real do usuário que o retângulo envolvendo Florianópolis, com erro máximo de apenas 20 metros. Além disso, graças ao plugin Google Activity Recognition, é possível identificar que o objeto móvel estava parado, com 100% de precisão. Com os dados coletados com os sensores magnetômetro e acelerômetro, é possível descobrir o azimute do objeto móvel para identificar a orientação do usuário, como mostra a figura 5.

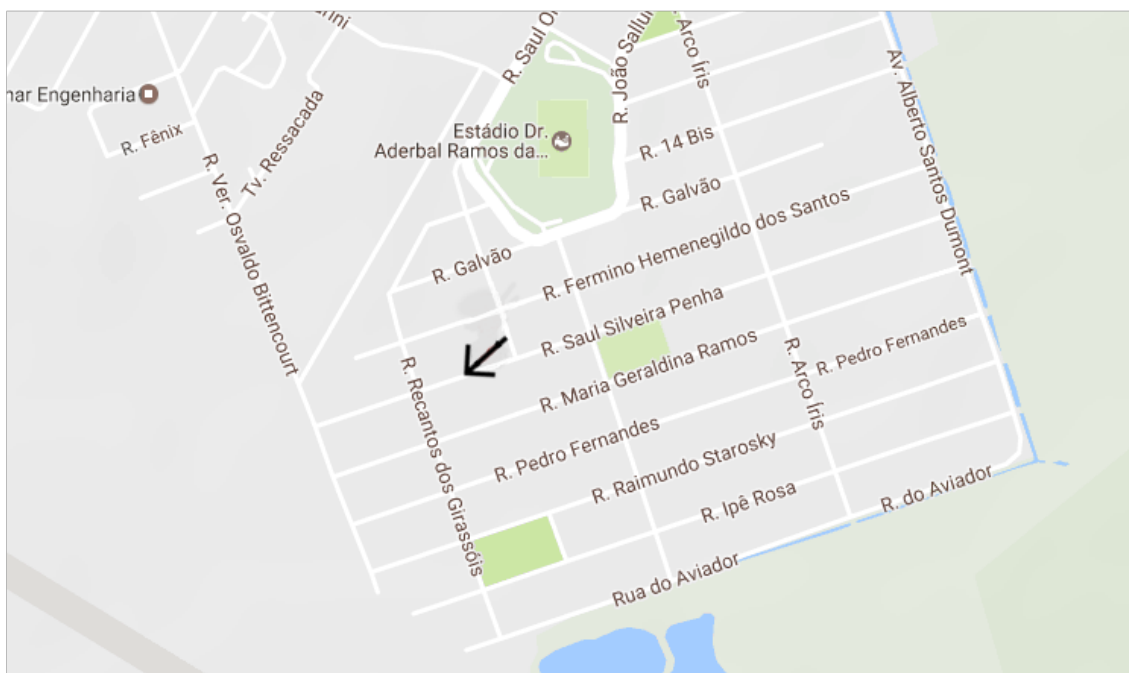


Figura 5. Mapa gerado pela API do google maps com a orientação do usuário

5.1. Teste de enriquecimento semântico com LOD

Para o teste de enriquecimento semântico LOD foi utilizado o tweet da figura 6. Este tweet, tal como o exemplo da figura 4, possui apenas identificação de local, que pode ser aproximado pelo polígono envolvendo Florianópolis. Também foi utilizado o algoritmo de ligação de postagens a locais proposto junto ao Baquara [Fileto et al. 2015].

Saindo da UFSC

18:16 - 30 de mai de 2017 de Florianópolis, Brasil



Figura 6. Tweet utilizado para o teste

Utilizando o Tweet Context Collector foi possível identificar a localização do objeto móvel no momento da postagem. A localização está representada na figura 7.

Também foi utilizado o recurso do dbpedia "Universidade Federal de Santa Catarina"¹³.



Figura 7. Tweet utilizado para o teste

O algoritmo do Baquara compara a distância entre o recurso e o objeto móvel e a similaridade do texto do tweet com o nome do recurso e exige valores mínimos e máximos para verificar se o recurso deve ser associado ao texto. Utilizamos uma distância máxima de 700m e uma similaridade mínima de 50%.

Como demonstrado na figura 8, a distância entre o objeto móvel e a UFSC é de 680,04 metros. Utilizando a métrica para comparação de strings Jaro-Winkler, descobrimos que a similaridade entre UFSC e o nome do recurso "Universidade Federal de Santa Catarina" é de 51,75%. Como tanto a distância é menor que a máxima, quanto a similaridade é maior que a mínima, o recurso serve para o enriquecimento semântico do tweet. No entanto, obviamente, um simples dicionário de nomes de superfície correlacionado tais strings resolveria o problema de casamento entre o nome por extenso e o acrônimo da UFSC.

Além disso, como o objeto móvel estava em movimento, o sensor de localização detectou que a velocidade era de 1.5 metros por segundo (5.4 km/h) e que a orientação era de 224.5 graus. Os dados retirados a partir do plugin Google Activity Recognition também acrescentam dados, indicando que o usuário estava se movimentando a pé.

¹³http://es.dbpedia.org/page/Universidad_Federal_de_Santa_Catarina

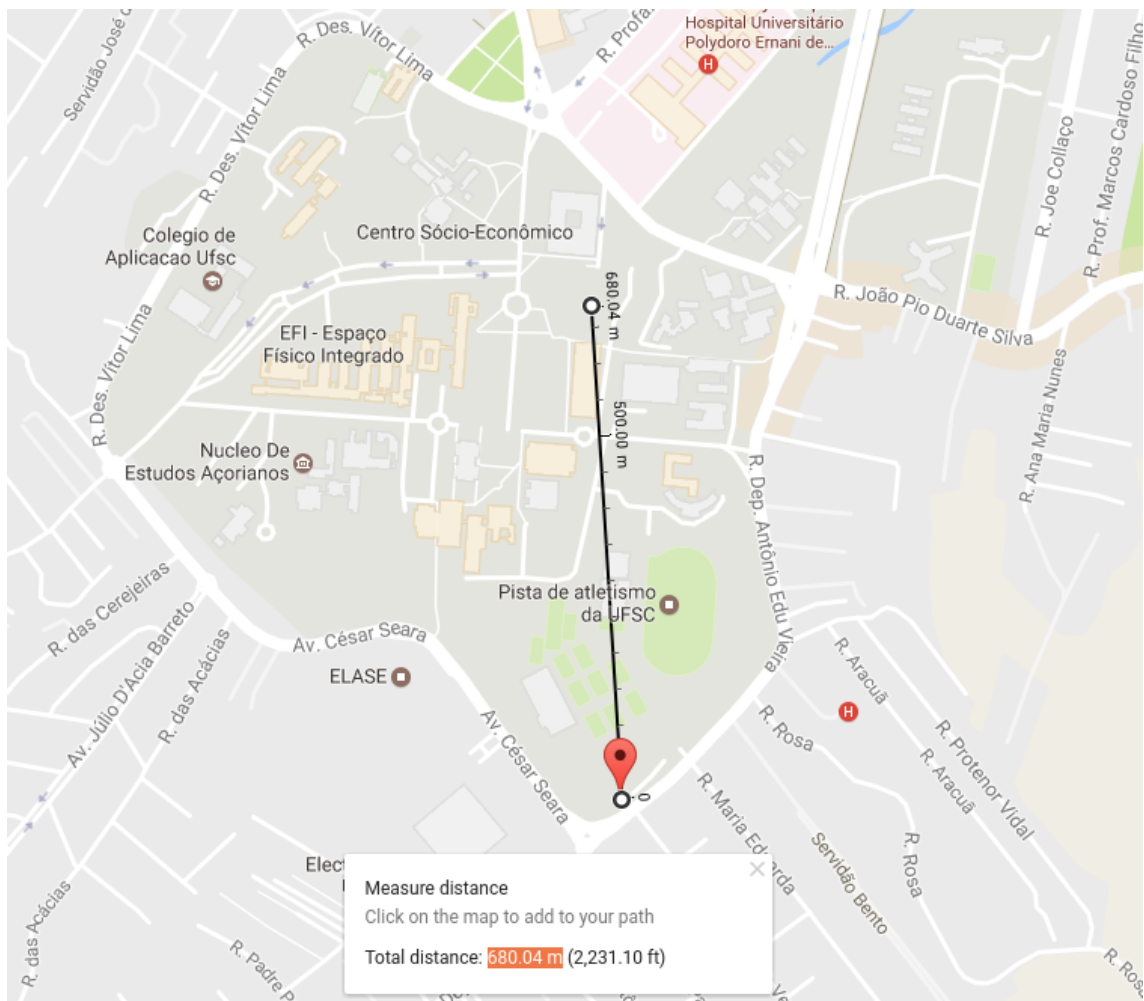


Figura 8. Distância calculada pela API do Google Maps

6. Conclusões e trabalhos futuros

Este trabalho apresentou uma revisão bibliográfica com os conceitos necessários ao entendimento de questões referentes ao enriquecimento semântico de dados sobre objetos móveis e o uso de informação de contexto em tal enriquecimento. Anotações semânticas, que são a base de tal enriquecimento, podem ser feitas e/ou sua qualidade aferida mediante o uso de dados sobre o contexto desses objetos móveis. Assim, este trabalho também apresentou um estudo do estado da arte sobre ferramentas de coleta de informação sobre o contexto de objetos móveis. Finalmente, apresentou um aplicativo desenvolvido para complementação de dados coletados de objetos móveis com informação de contexto e reportou alguns testes realizados para validar tal aplicativo. Assim, as contribuições deste trabalho são:

1. estudo comparativo de algumas das principais ferramentas para coleta de informação sobre o contexto de objetos móveis;
2. desenvolvimento de um aplicativo baseado em uma ferramenta de coleta de dados sobre o contexto para complementar dados coletados de mídias sociais com dados como coordenadas geográficas;

3. Um teste caracterizando dados coletados do Twitter e o uso do aplicativo para associar coordenadas geográficas a tweets que não as incluíam.

O teste mostrou que a maioria dos tweets coletados não possui coordenadas geográficas, que são necessárias a alguns processos de enriquecimento semântico. O aplicativo baseado em coleta de dados de contexto foi capaz de obter essas coordenadas e associar aos tweets que não as tinha para então realizar o processo de anotação semântica.

Para trabalhos futuros sugere-se:

1. pesquisar formas de aprimorar o Tweet Context Collector, diminuindo o consumo de bateria e conectando o com outros plugins;
2. realizar experimentos com mais voluntários e maiores amostras de tweets;
3. criar regras ouro para o enriquecimento semântico de tweets.

Referências

- Achrekar, H., Gandhe, A., Lazarus, R., Yu, S.-H., and Liu, B. (2011). Predicting flu trends using twitter data. *The First International Workshop on Cyber-Physical Networking Systems*.
- Agarwal, A., Xie, B., Vovsha, I., Rambow, O., and Passonneau, R. (2011). Sentiment analysis of twitter data. *LSM '11 Proceedings of the Workshop on Languages in Social Media*.
- Chen, Z., Fu, R., Zhao, Z., Cheng, P., Cao, C. C., Liu, Z., Tong, Y., Xia, L., Chen, L., and Zhang, C. J. (2010). gmission: A general spatial crowdsourcing platform. *ICTD '10 Proceedings of the 4th ACM/IEEE International Conference on Information and Communication Technologies and Development*.
- Chittilappilly, A. I., Chen, L., and Amer-Yahia, S. (2016). A survey of general-purpose crowdsourcing techniques. *IEEE Transactions on Knowledge and Data Engineering (Volume: 28, Issue: 9, Sept. 2016)*.
- de Sousa Duarte, P. A., Barreto, F. M., de Almada Gomes, F. A., de Carvalho, W. V., and Trinta, F. A. M. (2014). A model-driven approach to generate context aware applications. *WebMedia '14 Proceedings of the 20th Brazilian Symposium on Multimedia and the Web Pages 99-102*.
- Duarte, P. A. S., Silva, L. F. M. S., Gomes, F. A. A., Viana, W., and Trinta, F. A. M. (2015). Dynamic deployment for context-aware multimedia environments. *WebMedia '15 Proceedings of the 21st Brazilian Symposium on Multimedia and the Web Pages 197-204*.
- Ferreira, D., Kostakos, V., and Dey, A. K. (2014). Aware: mobile context instrumentation framework. *WebMedia '14 Proceedings of the 20th Brazilian Symposium on Multimedia and the Web Pages 99-102*.
- Fileto, R., May, C., Renso, C., Pelekis, N., Klein, D., and Theodoridis, Y. (2015). The baquara2 knowledge-based framework for semantic enrichment and analysis of movement data. *Data & Knowledge Engineering*.
- Frutuoso, D. G. (2014). Recuperação de informação e classificação de entidades organizacionais em textos não estruturados. Master's thesis, UFPE.

- Garritty, C. and Emam, K. E. (2006). Who's using pdas? estimates of pda use by health care providers: a systematic review of surveys.
- Hartung, C., Anokwa, Y., Brunette, W., Lerer, A., Tseng, C., and Borriello, G. (2010). Open data kit: Tools to build information services for developing regions. *ICTD '10 Proceedings of the 4th ACM/IEEE International Conference on Information and Communication Technologies and Development*.
- Kouloumpis, E., Wilson, T., and Moore, J. (2011). Twitter sentiment analysis: the good the bad and the omg! *Fifth International AAAI Conference on Weblogs and Social Media*.
- Kumar, S., Morstatter, F., and Liu, H. (2014). *Twitter Data Analytics*. Springer Briefs in Computer Science. Springer.
- Mathioudakis, M. and Koudas, N. (2010). Twittermonitor: Trend detection over the twitter stream. *International Conference on Management of data*.
- Oliveira, O., Bolliger, F., and Florido, A. (2006). Brazil agricultural census 2006: Innovations and impacts.
- Paul, M. J. and Dredze, M. (2011). You are what you tweet: Analyzing twitter for public health. *Fifth International AAAI Conference on Weblogs and Social Media*.
- Wu, F., Li, Z., Lee, W.-C., Wang, H., and Huang, Z. (2015). Semantic annotation of mobility data using social media. *24th International Conference on World Wide Web*.

APÊNDICE B – Modelo do banco de dados para experimentos de enri- quecimento semântico

Listing B.1 – code/db.sql

```

1  -- MySQL Script generated by MySQL Workbench
2  -- Qui 06 Jul 2017 00:10:55 BRT
3  -- Model: New Model      Version: 1.0
4  -- MySQL Workbench Forward Engineering
5
6  SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
7  SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
    FOREIGN_KEY_CHECKS=0;
8  SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,
    ALLOW_INVALID_DATES';
9
10 -----
11 -- Schema mydb
12 -----
13
14 -----
15 -- Schema mydb
16 -----
17 CREATE SCHEMA IF NOT EXISTS 'mydb' DEFAULT CHARACTER SET utf8 ;
18 USE 'mydb' ;
19
20 -----
21 -- Table 'mydb`.`state`
22 -----
23 CREATE TABLE IF NOT EXISTS 'mydb`.`state` (
24 )
25 ENGINE = InnoDB;
26
27
28 -----
29 -- Table 'mydb`.`DataSource`
30 -----
31 CREATE TABLE IF NOT EXISTS 'mydb`.`DataSource` (
32   'idDSource' INT NOT NULL,
33   'name' VARCHAR(30) NOT NULL,

```

```
34     'url' VARCHAR(45) NULL ,
35     PRIMARY KEY ('idDSource'))
36 ENGINE = InnoDB;
37
38
39 -----
40 -- Table 'mydb`.`DataSet`
41 -----
42 CREATE TABLE IF NOT EXISTS 'mydb`.`DataSet` (
43     'idDSet' INT NOT NULL ,
44     'name' VARCHAR(30) NOT NULL ,
45     'idSource' INT NOT NULL ,
46     'dateBeginGathering' DATE NOT NULL ,
47     'dateEndGathering' DATE NOT NULL ,
48     PRIMARY KEY ('idDSet'),
49     INDEX 'fk_Movement Dataset_Source1_idx' ('idSource' ASC),
50     CONSTRAINT 'fk_Movement Dataset_Source1`
51         FOREIGN KEY ('idSource')
52         REFERENCES 'mydb`.`DataSource` ('idDSource')
53         ON DELETE NO ACTION
54         ON UPDATE NO ACTION)
55 ENGINE = InnoDB;
56
57
58 -----
59 -- Table 'mydb`.`User`
60 -----
61 CREATE TABLE IF NOT EXISTS 'mydb`.`User` (
62     'idUser' BIGINT NOT NULL ,
63     'idDSource ?' INT NOT NULL ,
64     'idO' BIGINT NOT NULL ,
65     'screenname' VARCHAR(15) NULL COMMENT 'The screen name, handle,
66         or alias that this user identifies themselves with.
67         screen_names are unique but subject to change. ',
68     'name' VARCHAR(20) NULL COMMENT 'The name of the user, as theyve
69         defined it. Not necessarily a persons name.\n',
70     'location' VARCHAR(70) NULL COMMENT 'The user-defined location
71         for this accounts profile. Not necessarily a location nor
72         parseable',
73     'description' VARCHAR(160) NULL COMMENT 'The user-defined UTF-8
74         string describing their account.',
75     'isprotected' TINYINT(1) NULL COMMENT 'When true, indicates that
76         this user has chosen to protect their Tweets',
77     'followerscount' INT NULL COMMENT 'The number of followers this
78         account currently has. Under certain conditions of duress,
79         this field will temporarily indicate 0',
80     'friendscount' INT NULL COMMENT 'The number of users this
```

```

    account is following',
72 'createdate' DATETIME NULL COMMENT 'The UTC datetime that the
    user account was created on Twitter.\n',
73 'favoritecount' INT NULL COMMENT 'The number of tweets this user
    has favorited in the accounts lifetime. ',
74 'utcoffset' INT NULL COMMENT 'The offset from GMT/UTC in seconds
    .',
75 'timezone' VARCHAR(50) NULL COMMENT ' A string describing the
    Time Zone this user declares themselves within.',
76 'lang' VARCHAR(6) NULL COMMENT 'The BCP 47 code for the users
    self-declared user interface language',
77 'postscout' INT NULL COMMENT 'The number of tweets (including
    retweets) issued by the user.',
78 'isgeoenabled' TINYINT(1) NULL COMMENT 'When true, indicates
    that the user has enabled the possibility of geotagging their
    Tweets.',
79 'isverified' TINYINT(1) NULL COMMENT 'When true, indicates that
    the user has a verified account',
80 'istranslator' TINYINT(1) NULL COMMENT 'When true, indicates
    that the user is a participant in Twitters translator
    community',
81 'listedcount' INT NULL COMMENT 'The number of public lists that
    this user is a member of.',
82 'url' VARCHAR(60) NULL COMMENT 'A URL provided by the user in
    association with their profile.\n',
83 PRIMARY KEY ('idUser'),
84 INDEX 'fk_User_DataSource_idx' ('idDSource' ?' ASC),
85 INDEX 'fk_User_Object_idx' ('idO' ASC),
86 CONSTRAINT 'fk_User_DataSource'
87     FOREIGN KEY ('idDSource' ?')
88     REFERENCES 'mydb'.'DataSource' ('idDSource' )
89     ON DELETE NO ACTION
90     ON UPDATE NO ACTION,
91 CONSTRAINT 'fk_User_Object'
92     FOREIGN KEY ('idO')
93     REFERENCES 'mydb'.'Object' ('idO')
94     ON DELETE NO ACTION
95     ON UPDATE NO ACTION)
96 ENGINE = InnoDB;
97
98
99 -----
100 -- Table 'mydb'.'Object'
101 -----
102 CREATE TABLE IF NOT EXISTS 'mydb'.'Object' (
103     'idDSource' INT NOT NULL,
104     'idO' BIGINT NOT NULL,
```

```

105 INDEX 'fk_Moving_Object_user1_idx' ('idO' ASC),
106 PRIMARY KEY ('idDSource', 'idO'),
107 INDEX 'fk_Object_MDSource1_idx' ('idDSource' ASC),
108 CONSTRAINT 'fk_Moving_Object_user1'
109 FOREIGN KEY ('idO')
110 REFERENCES 'mydb'.'User' ('idO')
111 ON DELETE NO ACTION
112 ON UPDATE NO ACTION,
113 CONSTRAINT 'fk_Object_MDSource1'
114 FOREIGN KEY ('idDSource')
115 REFERENCES 'mydb'.'DataSource' ('idDSource')
116 ON DELETE RESTRICT
117 ON UPDATE RESTRICT)
118 ENGINE = InnoDB;
119
120
121 -----
122 -- Table 'mydb'.'Position'
123 -----
124 CREATE TABLE IF NOT EXISTS 'mydb'.'Position' (
125 'idDSet' INT NOT NULL,
126 'idO' INT NOT NULL,
127 'idPosition' BIGINT NOT NULL,
128 'timestamp' DATETIME NOT NULL,
129 'geom' POINT NULL,
130 PRIMARY KEY ('idDSet', 'idO', 'idPosition'),
131 INDEX 'fk_Position_MDSet1_idx' ('idDSet' ASC),
132 INDEX 'fk_Position_Moving_Object1_idx' ('idO' ASC),
133 CONSTRAINT 'fk_Position_MDSet1'
134 FOREIGN KEY ('idDSet')
135 REFERENCES 'mydb'.'DataSet' ('idDSet')
136 ON DELETE NO ACTION
137 ON UPDATE NO ACTION,
138 CONSTRAINT 'fk_Position_Moving_Object1'
139 FOREIGN KEY ('idO')
140 REFERENCES 'mydb'.'Object' ('idO')
141 ON DELETE RESTRICT
142 ON UPDATE RESTRICT)
143 ENGINE = InnoDB
144 COMMENT = ' ';
145
146
147 -----
148 -- Table 'mydb'.'resource'
149 -----
150 CREATE TABLE IF NOT EXISTS 'mydb'.'resource' (
151 'idresource' INT NOT NULL,

```



```
152     'resourcecol' VARCHAR(45) NULL ,
153     PRIMARY KEY ('idresource'))
154 ENGINE = InnoDB;
155
156
157 -----
158 -- Table 'mydb`.`KSource`
159 -----
160 CREATE TABLE IF NOT EXISTS 'mydb`.`KSource` (
161     'idKSource' INT NOT NULL ,
162     'name' VARCHAR(30) NOT NULL ,
163     'dateOfGathering' DATETIME NOT NULL ,
164     'version' VARCHAR(45) NULL ,
165     PRIMARY KEY ('idKSource'))
166 ENGINE = InnoDB;
167
168
169 -----
170 -- Table 'mydb`.`Resource`
171 -----
172 CREATE TABLE IF NOT EXISTS 'mydb`.`Resource` (
173     'idKSource' INT NOT NULL ,
174     'idResource' INT NOT NULL ,
175     'uri' VARCHAR(150) NOT NULL ,
176     PRIMARY KEY ('idKSource', 'idResource'),
177     INDEX 'fk_resource_source_idx' ('idKSource' ASC),
178     CONSTRAINT 'fk_resource_source`
179         FOREIGN KEY ('idKSource`)
180         REFERENCES 'mydb`.`KSource` ('idKSource`)
181         ON DELETE NO ACTION
182         ON UPDATE NO ACTION)
183 ENGINE = InnoDB;
184
185
186 -----
187 -- Table 'mydb`.`Label`
188 -----
189 CREATE TABLE IF NOT EXISTS 'mydb`.`Label` (
190     'idResource' INT NOT NULL ,
191     'idLabel' INT NOT NULL ,
192     'value' VARCHAR(150) NOT NULL ,
193     PRIMARY KEY ('idLabel', 'idResource'),
194     INDEX 'fk_label_resource1_idx' ('idResource' ASC),
195     CONSTRAINT 'fk_label_resource1`
196         FOREIGN KEY ('idResource`)
197         REFERENCES 'mydb`.`Resource` ('idResource`)
198         ON DELETE NO ACTION
```

```
199         ON UPDATE NO ACTION)
200 ENGINE = InnoDB;
201
202
203 -----
204 -- Table 'mydb`.`Method`
205 -----
206 CREATE TABLE IF NOT EXISTS 'mydb`.`Method` (
207     'idMethod' INT NOT NULL,
208     'name' VARCHAR(45) NULL,
209     'description' TEXT NULL,
210     'version' VARCHAR(45) NULL,
211     PRIMARY KEY ('idMethod'))
212 ENGINE = InnoDB;
213
214
215 -----
216 -- Table 'mydb`.`Person`
217 -----
218 CREATE TABLE IF NOT EXISTS 'mydb`.`Person` (
219     'idPerson' INT NOT NULL,
220     'nome' VARCHAR(25) NOT NULL,
221     'cargo' CHAR NOT NULL COMMENT 'U = undergraduate student; M =
222         Masters student; D - Doctorate student; P = Posdoc; A =
223         Academic Staff (professor, scholar, etc.); T = Technical
224         staff; V = Visitor; O = Other\n',
225     PRIMARY KEY ('idPerson'))
226 ENGINE = InnoDB;
227
228
229 -----
230 -- Table 'mydb`.`Experiment`
231 -----
232 CREATE TABLE IF NOT EXISTS 'mydb`.`Experiment` (
233     'idMethod' INT NOT NULL,
234     'idExperiment' INT NOT NULL AUTO_INCREMENT,
235     'beginTime' DATETIME NOT NULL,
236     'endTime' DATETIME NOT NULL,
237     'idRespPerson' INT NOT NULL,
238     PRIMARY KEY ('idMethod', 'idExperiment'),
239     INDEX 'fk_experiment_Method1_idx' ('idMethod' ASC),
240     INDEX 'fk_Experiment_Responsavel1_idx' ('idRespPerson' ASC),
241     CONSTRAINT 'fk_experiment_Method1'
242         FOREIGN KEY ('idMethod')
243         REFERENCES 'mydb`.`Method` ('idMethod')
244         ON DELETE RESTRICT
245         ON UPDATE RESTRICT,
```

```
243     CONSTRAINT 'fk_Experiment_Responsavel1'
244         FOREIGN KEY ('idRespPerson')
245         REFERENCES 'mydb'.'Person' ('idPerson')
246         ON DELETE RESTRICT
247         ON UPDATE RESTRICT)
248 ENGINE = InnoDB
249 COMMENT = ' ';
250
251
252 -----
253 -- Table 'mydb'.'Segment'
254 -----
255 CREATE TABLE IF NOT EXISTS 'mydb'.'Segment' (
256     'idDSet' INT NOT NULL,
257     'idO' INT NOT NULL,
258     'idSegment' INT NOT NULL,
259     'idInitialPosition' BIGINT NOT NULL,
260     'idFinalPosition' BIGINT NOT NULL,
261     'predicate' VARCHAR(45) NULL,
262     PRIMARY KEY ('idDSet', 'idO', 'idSegment'),
263     INDEX 'fk_Segment_position1_idx' ('idInitialPosition' ASC),
264     INDEX 'fk_Segment_position2_idx' ('idFinalPosition' ASC),
265     INDEX 'fk_Segment_Movement_Dataset1_idx' ('idDSet' ASC),
266     INDEX 'fk_Segment_Moving_Object1_idx' ('idO' ASC),
267     CONSTRAINT 'fk_Segment_position1'
268         FOREIGN KEY ('idInitialPosition')
269         REFERENCES 'mydb'.'Position' ('idPosition')
270         ON DELETE NO ACTION
271         ON UPDATE NO ACTION,
272     CONSTRAINT 'fk_Segment_position2'
273         FOREIGN KEY ('idFinalPosition')
274         REFERENCES 'mydb'.'Position' ('idPosition')
275         ON DELETE NO ACTION
276         ON UPDATE NO ACTION,
277     CONSTRAINT 'fk_Segment_Movement_Dataset1'
278         FOREIGN KEY ('idDSet')
279         REFERENCES 'mydb'.'DataSet' ('idDSet')
280         ON DELETE NO ACTION
281         ON UPDATE NO ACTION,
282     CONSTRAINT 'fk_Segment_Moving_Object1'
283         FOREIGN KEY ('idO')
284         REFERENCES 'mydb'.'Object' ('idO')
285         ON DELETE NO ACTION
286         ON UPDATE NO ACTION)
287 ENGINE = InnoDB;
288
289
```

```
290 |-----
291 |-- Table 'mydb'.'SegmAssociation'
292 |-----
293 |CREATE TABLE IF NOT EXISTS 'mydb'.'SegmAssociation' (
294 |  'idMethod' INT NOT NULL,
295 |  'idExperiment' INT NOT NULL,
296 |  'idSegment' INT NOT NULL,
297 |  'idResource' INT NOT NULL,
298 |  'score' FLOAT NULL,
299 |  PRIMARY KEY ('idMethod', 'idExperiment', 'idSegment', '
      |      idResource'),
300 |  INDEX 'fk_match_resource1_idx' ('idResource' ASC),
301 |  INDEX 'fk_association_Experiment1_idx' ('idMethod' ASC, '
      |      idExperiment' ASC),
302 |  INDEX 'fk_association_Segment1_idx' ('idSegment' ASC),
303 |  CONSTRAINT 'fk_match_resource1'
304 |    FOREIGN KEY ('idResource')
305 |    REFERENCES 'mydb'.'Resource' ('idResource')
306 |    ON DELETE NO ACTION
307 |    ON UPDATE NO ACTION,
308 |  CONSTRAINT 'fk_association_Experiment1'
309 |    FOREIGN KEY ('idMethod', 'idExperiment')
310 |    REFERENCES 'mydb'.'Experiment' ('idMethod', 'idExperiment')
311 |    ON DELETE RESTRICT
312 |    ON UPDATE RESTRICT,
313 |  CONSTRAINT 'fk_association_Segment1'
314 |    FOREIGN KEY ('idSegment')
315 |    REFERENCES 'mydb'.'Segment' ('idSegment')
316 |    ON DELETE NO ACTION
317 |    ON UPDATE NO ACTION)
318 |ENGINE = InnoDB;
319 |
320 |
321 |-----
322 |-- Table 'mydb'.'Resource_Type'
323 |-----
324 |CREATE TABLE IF NOT EXISTS 'mydb'.'Resource_Type' (
325 |  'idInstace' INT NOT NULL,
326 |  'idType' INT NOT NULL,
327 |  PRIMARY KEY ('idInstace', 'idType'),
328 |  INDEX 'fk_resource_type_resource1_idx' ('idInstace' ASC),
329 |  INDEX 'fk_resource_type_resourceInstance1_idx' ('idType' ASC),
330 |  CONSTRAINT 'fk_resource_type_resource1'
331 |    FOREIGN KEY ('idInstace')
332 |    REFERENCES 'mydb'.'Resource' ('idResource')
333 |    ON DELETE RESTRICT
334 |    ON UPDATE RESTRICT,
```

```
335     CONSTRAINT 'fk_resource_type_resourceInstance1'
336         FOREIGN KEY ('idType')
337         REFERENCES 'mydb`.`Resource' ('idResource')
338         ON DELETE RESTRICT
339         ON UPDATE RESTRICT)
340 ENGINE = InnoDB;
341
342
343 -----
344 -- Table 'mydb`.`Place'
345 -----
346 CREATE TABLE IF NOT EXISTS 'mydb`.`Place' (
347     'idPlace' VARCHAR(16) NOT NULL,
348     'name' VARCHAR(60) NULL COMMENT 'Short human-readable
349         representation of the places name.',
350     'fullname' VARCHAR(80) NULL COMMENT 'Full human-readable
351         representation of the places name.',
352     'countrycode' VARCHAR(2) NULL COMMENT 'Shortened country code
353         representing the country containing this place.',
354     'country' VARCHAR(40) NULL COMMENT 'Name of the country
355         containing this place.',
356     'placetype' VARCHAR(12) NULL COMMENT 'The type of location
357         represented by this place.',
358     'url' VARCHAR(60) NULL COMMENT 'URL representing the location of
359         additional place metadata for this place.',
360     'bbminlon' DOUBLE NULL COMMENT 'A bounding box of coordinates
361         which encloses this place. min longitude\n',
362     'bbminlat' DOUBLE NULL COMMENT 'A bounding box of coordinates
363         which encloses this place. min latitude\n',
364     'bbmaxlon' DOUBLE NULL COMMENT 'A bounding box of coordinates
365         which encloses this place. max longitude\n',
366     'bbmaxlat' DOUBLE NULL COMMENT 'A bounding box of coordinates
367         which encloses this place. max latitude\n',
368     PRIMARY KEY ('idPlace'))
369 ENGINE = InnoDB;
370
371 -----
372 -- Table 'mydb`.`Post'
373 -----
374 CREATE TABLE IF NOT EXISTS 'mydb`.`Post' (
375     'idPost' BIGINT NOT NULL,
376     'idUser' BIGINT NOT NULL COMMENT 'The user who posted this Tweet
377         ',
378     'idDSet' INT NOT NULL,
379     'idO' BIGINT NOT NULL,
380     'idPosition' BIGINT NOT NULL,
```

```
371 | 'idRepost' BIGINT NULL COMMENT 'Users can amplify the broadcast
      | of tweets authored by other users by retweeting.',
372 | 'idPlace' VARCHAR(20) NULL COMMENT 'When present, indicates that
      | the tweet is associated',
373 | 'idOrigSource' INT NULL COMMENT 'Utility used to post the Tweet\
      | nCould it be an idMDSource?',
374 | 'createdate' DATETIME NULL COMMENT 'UTC time when this Tweet was
      | created.',
375 | 'text' VARCHAR(140) NOT NULL COMMENT 'The actual UTF-8 text of
      | the status update',
376 | 'istruncated' TINYINT(1) NULL COMMENT 'Indicates whether the
      | value of the text parameter was truncated, for example, as a
      | result of a retweet exceeding the 140 character Tweet length.
      | ',
377 | 'inreplytostatusid' BIGINT NULL COMMENT 'If the represented
      | Tweet is a reply, this field will contain the integer
      | representation of the original Tweets ID.',
378 | 'repostcount' INT NULL COMMENT 'Number of times this Tweet has
      | been retweeted',
379 | 'likecount' INT NULL COMMENT 'Indicates approximately how many
      | times this Tweet has been liked by Twitter users.',
380 | 'ispossiblyensitive' TINYINT(1) NULL COMMENT 'This field only
      | surfaces when a tweet contains a link. The meaning of the
      | field doesnt pertain to the tweet content itself, but instead
      | it is an indicator that the URL contained in the tweet may
      | contain content or media identified as sensitive content.',
381 | 'lang' VARCHAR(6) NULL COMMENT 'When present, indicates a BCP 47
      | language identifier corresponding to the machine-detected
      | language of the Tweet text, or und if no language could be
      | detected.',
382 | PRIMARY KEY ('idPost'),
383 | INDEX 'fk_tweet_position1_idx' (('idPosition' ASC),
384 | INDEX 'fk_tweet_tweet1_idx' (('idRepost' ASC),
385 | INDEX 'fk_tweet_place_idx' (('idPlace' ASC),
386 | INDEX 'fk_post_user_idx' (('idUser' ASC),
387 | INDEX 'fk_Post_object_idx' (('idO' ASC),
388 | INDEX 'fk_post_datasry_idx' (('idDSet' ASC),
389 | INDEX 'fk_post_datasource_idx' (('idOrigSource' ASC),
390 | CONSTRAINT 'fk_post_position'
391 | FOREIGN KEY ('idPosition')
392 | REFERENCES 'mydb'..'Position' (('idPosition')
393 | ON DELETE NO ACTION
394 | ON UPDATE NO ACTION,
395 | CONSTRAINT 'fk_post_post'
396 | FOREIGN KEY ('idRepost')
397 | REFERENCES 'mydb'..'Post' (('idPost')
398 | ON DELETE NO ACTION
```

```
399     ON UPDATE NO ACTION ,
400 CONSTRAINT 'fk_post_place'
401     FOREIGN KEY ('idPlace')
402     REFERENCES 'mydb'.'Place' ('idPlace')
403     ON DELETE NO ACTION
404     ON UPDATE NO ACTION ,
405 CONSTRAINT 'fk_post_user'
406     FOREIGN KEY ('idUser')
407     REFERENCES 'mydb'.'User' ('idUser')
408     ON DELETE NO ACTION
409     ON UPDATE NO ACTION ,
410 CONSTRAINT 'fk_post_datasry'
411     FOREIGN KEY ('idDSet')
412     REFERENCES 'mydb'.'DataSet' ('idDSet')
413     ON DELETE NO ACTION
414     ON UPDATE NO ACTION ,
415 CONSTRAINT 'fk_Post_object'
416     FOREIGN KEY ('idO')
417     REFERENCES 'mydb'.'Object' ('idO')
418     ON DELETE NO ACTION
419     ON UPDATE NO ACTION ,
420 CONSTRAINT 'fk_post_datasource'
421     FOREIGN KEY ('idOrigSource')
422     REFERENCES 'mydb'.'DataSource' ('idDSource')
423     ON DELETE NO ACTION
424     ON UPDATE NO ACTION)
425 ENGINE = InnoDB;
426
427
428 -----
429 -- Table 'mydb'.'RelevantWord'
430 -----
431 CREATE TABLE IF NOT EXISTS 'mydb'.'RelevantWord' (
432     'idRW' INT NOT NULL ,
433     'RWcategory' CHAR NOT NULL COMMENT 'Hashtag, NamedEntity,
434         Mofological\n',
435     'RWvalue' VARCHAR(45) NOT NULL ,
436     'RWinitialPosText' INT NULL ,
437     'RWfinalPosText' INT NULL ,
438     PRIMARY KEY ('idRW'))
439 ENGINE = InnoDB;
440
441 -----
442 -- Table 'mydb'.'MentionsUser'
443 -----
444 CREATE TABLE IF NOT EXISTS 'mydb'.'MentionsUser' (
```

```
445     'idPost' BIGINT NOT NULL,
446     'idUser' BIGINT NOT NULL,
447     PRIMARY KEY ('idPost', 'idUser'),
448     INDEX 'fk_tweet_has_user_user1_idx' ('idUser' ASC),
449     INDEX 'fk_mentionedUser_tweet1_idx' ('idPost' ASC),
450     CONSTRAINT 'fk_tweet_has_user_user1'
451         FOREIGN KEY ('idUser')
452         REFERENCES 'mydb`.`User' ('id0')
453         ON DELETE NO ACTION
454         ON UPDATE NO ACTION,
455     CONSTRAINT 'fk_mentionedUser_tweet1'
456         FOREIGN KEY ('idPost')
457         REFERENCES 'mydb`.`Post' ('idPost')
458         ON DELETE NO ACTION
459         ON UPDATE NO ACTION)
460 ENGINE = InnoDB;
461
462
463 -----
464 -- Table 'mydb`.`Post_has_RW'
465 -----
466 CREATE TABLE IF NOT EXISTS 'mydb`.`Post_has_RW' (
467     'idPost' BIGINT NOT NULL,
468     'idRW' INT NOT NULL,
469     PRIMARY KEY ('idPost', 'idRW'),
470     INDEX 'fk_tweet_has_hashtag_hashtag1_idx' ('idRW' ASC),
471     INDEX 'fk_tweet_has_hashtag_tweet1_idx' ('idPost' ASC),
472     CONSTRAINT 'fk_tweet_has_hashtag_tweet1'
473         FOREIGN KEY ('idPost')
474         REFERENCES 'mydb`.`Post' ('idPost')
475         ON DELETE NO ACTION
476         ON UPDATE NO ACTION,
477     CONSTRAINT 'fk_tweet_has_hashtag_hashtag1'
478         FOREIGN KEY ('idRW')
479         REFERENCES 'mydb`.`RelevantWord' ('idRW')
480         ON DELETE NO ACTION
481         ON UPDATE NO ACTION)
482 ENGINE = InnoDB
483 COMMENT = 'Post has relevant word';
484
485
486 -----
487 -- Table 'mydb`.`Geometry'
488 -----
489 CREATE TABLE IF NOT EXISTS 'mydb`.`Geometry' (
490     'idGeometry' INT NOT NULL,
491     'geom' GEOMETRYCOLLECTION NOT NULL,
```



```
492     PRIMARY KEY ('idGeometry'))
493 ENGINE = InnoDB;
494
495
496 -----
497 -- Table 'mydb`.`Resource_has_Geom`
498 -----
499 CREATE TABLE IF NOT EXISTS 'mydb`.`Resource_has_Geom` (
500     'idResource' INT NOT NULL,
501     'idGeometry' INT NOT NULL,
502     PRIMARY KEY ('idResource', 'idGeometry'),
503     INDEX 'fk_resource_has_location_location1_idx' ('idGeometry' ASC
504         ),
505     INDEX 'fk_resource_has_location_resource1_idx' ('idResource' ASC
506         ),
507     CONSTRAINT 'fk_resource_has_location_resource1'
508         FOREIGN KEY ('idResource')
509         REFERENCES 'mydb`.`Resource` ('idResource')
510         ON DELETE NO ACTION
511         ON UPDATE NO ACTION,
512     CONSTRAINT 'fk_resource_has_location_location1'
513         FOREIGN KEY ('idGeometry')
514         REFERENCES 'mydb`.`Geometry` ('idGeometry')
515         ON DELETE NO ACTION
516         ON UPDATE NO ACTION)
517 ENGINE = InnoDB;
518
519 -----
520 -- Table 'mydb`.`Parameter`
521 -----
522 CREATE TABLE IF NOT EXISTS 'mydb`.`Parameter` (
523     'idMethod' INT NOT NULL,
524     'idParameter' INT NOT NULL,
525     'name' VARCHAR(45) NULL,
526     PRIMARY KEY ('idMethod', 'idParameter'),
527     INDEX 'fk_Parameter_Method1_idx' ('idMethod' ASC),
528     CONSTRAINT 'fk_Parameter_Method1'
529         FOREIGN KEY ('idMethod')
530         REFERENCES 'mydb`.`Method` ('idMethod')
531         ON DELETE RESTRICT
532         ON UPDATE RESTRICT)
533 ENGINE = InnoDB;
534
535 -----
536 -- Table 'mydb`.`Exper_uses_Param`
```

```
537 |-----
538 | CREATE TABLE IF NOT EXISTS 'mydb'.'Exper_uses_Param' (
539 |   'idMethod' INT NOT NULL,
540 |   'idExperiment' INT NOT NULL,
541 |   'idParameter' INT NOT NULL,
542 |   'value' FLOAT NOT NULL,
543 |   PRIMARY KEY ('idMethod', 'idExperiment', 'idParameter'),
544 |   INDEX 'fk_Parameter_has_Experiment_Experiment1_idx' ('
545 |     idExperiment' ASC),
546 |   INDEX 'fk_Parameter_has_Experiment_Parameter1_idx' ('idMethod'
547 |     ASC, 'idParameter' ASC),
548 |   CONSTRAINT 'fk_Parameter_has_Experiment_Parameter1'
549 |     FOREIGN KEY ('idMethod', 'idParameter')
550 |     REFERENCES 'mydb'.'Parameter' ('idMethod', 'idParameter')
551 |     ON DELETE RESTRICT
552 |     ON UPDATE RESTRICT,
553 |   CONSTRAINT 'fk_Parameter_has_Experiment_Experiment1'
554 |     FOREIGN KEY ('idExperiment')
555 |     REFERENCES 'mydb'.'Experiment' ('idExperiment')
556 |     ON DELETE RESTRICT
557 |     ON UPDATE RESTRICT)
558 | ENGINE = InnoDB;
559 |-----
560 | -- Table 'mydb'.'Segment_has_SubSegment'
561 |-----
562 | CREATE TABLE IF NOT EXISTS 'mydb'.'Segment_has_SubSegment' (
563 |   'idDSet' INT NOT NULL,
564 |   'idMO' INT NOT NULL,
565 |   'idSuperSegm' INT NOT NULL,
566 |   'idSubSegm' INT NOT NULL,
567 |   PRIMARY KEY ('idDSet', 'idMO', 'idSuperSegm', 'idSubSegm'),
568 |   INDEX 'fk_Segment_has_Segment_Segment2_idx' ('idSubSegm' ASC),
569 |   INDEX 'fk_Segment_has_Segment_Segment1_idx' ('idDSet' ASC, 'idMO
570 |     ' ASC, 'idSuperSegm' ASC),
571 |   CONSTRAINT 'fk_Segment_has_Segment_Segment1'
572 |     FOREIGN KEY ('idDSet', 'idMO', 'idSuperSegm')
573 |     REFERENCES 'mydb'.'Segment' ('idDSet', 'idMO', 'idSegment')
574 |     ON DELETE NO ACTION
575 |     ON UPDATE NO ACTION,
576 |   CONSTRAINT 'fk_Segment_has_Segment_Segment2'
577 |     FOREIGN KEY ('idSubSegm')
578 |     REFERENCES 'mydb'.'Segment' ('idSegment')
579 |     ON DELETE NO ACTION
580 |     ON UPDATE NO ACTION)
581 | ENGINE = InnoDB;
```

```
581
582
583 -----
584 -- Table 'mydb`.`table1`
585 -----
586 CREATE TABLE IF NOT EXISTS 'mydb`.`table1` (
587 )
588 ENGINE = InnoDB;
589
590
591 -----
592 -- Table 'mydb`.`RWAssociation`
593 -----
594 CREATE TABLE IF NOT EXISTS 'mydb`.`RWAssociation` (
595   'idMethod' INT NOT NULL,
596   'idExperiment' INT NOT NULL,
597   'idPost' BIGINT NOT NULL,
598   'idRW' INT NOT NULL,
599   'idSource' INT NOT NULL,
600   'idResource' INT NOT NULL,
601   PRIMARY KEY ('idPost', 'idRW', 'idMethod', 'idExperiment', '
        idSource', 'idResource'),
602   INDEX 'fk_Experiment_has_RelevantWord_RelevantWord1_idx' ('idRW'
        ASC),
603   INDEX 'fk_Experiment_has_RelevantWord_Experiment1_idx' ('
        idMethod' ASC, 'idExperiment' ASC),
604   INDEX 'fk_RWAssociation_Resource1_idx' ('idSource' ASC, '
        idResource' ASC),
605   CONSTRAINT 'fk_Experiment_has_RelevantWord_Experiment1`
606     FOREIGN KEY ('idMethod', 'idExperiment')
607     REFERENCES 'mydb`.`Experiment` ('idMethod', 'idExperiment')
608     ON DELETE RESTRICT
609     ON UPDATE RESTRICT,
610   CONSTRAINT 'fk_Experiment_has_RelevantWord_RelevantWord1`
611     FOREIGN KEY ('idRW')
612     REFERENCES 'mydb`.`RelevantWord` ('idRW')
613     ON DELETE RESTRICT
614     ON UPDATE RESTRICT,
615   CONSTRAINT 'fk_RWAssociation_Resource1`
616     FOREIGN KEY ('idSource', 'idResource')
617     REFERENCES 'mydb`.`Resource` ('idKSource', 'idResource')
618     ON DELETE RESTRICT
619     ON UPDATE RESTRICT)
620 ENGINE = InnoDB;
621
622
623 -----
```

```
624 -- Table 'mydb'.'Experiment_extracts_RW'
625 -----
626 CREATE TABLE IF NOT EXISTS 'mydb'.'Experiment_extracts_RW' (
627   'idMethod' INT NOT NULL,
628   'idExperiment' INT NOT NULL,
629   'idPost' BIGINT NOT NULL,
630   'idRW' INT NOT NULL,
631   PRIMARY KEY ('idMethod', 'idExperiment', 'idPost', 'idRW'),
632   INDEX 'fk_Experiment_has_Post_has_RW_Post_has_RW1_idx' ('idPost'
        ASC, 'idRW' ASC),
633   INDEX 'fk_Experiment_has_Post_has_RW_Experiment1_idx' ('idMethod
        ASC, 'idExperiment' ASC),
634   CONSTRAINT 'fk_Experiment_has_Post_has_RW_Experiment1'
635     FOREIGN KEY ('idMethod', 'idExperiment')
636     REFERENCES 'mydb'.'Experiment' ('idMethod', 'idExperiment')
637     ON DELETE RESTRICT
638     ON UPDATE RESTRICT,
639   CONSTRAINT 'fk_Experiment_has_Post_has_RW_Post_has_RW1'
640     FOREIGN KEY ('idPost', 'idRW')
641     REFERENCES 'mydb'.'Post_has_RW' ('idPost', 'idRW')
642     ON DELETE RESTRICT
643     ON UPDATE RESTRICT)
644 ENGINE = InnoDB;
645
646
647 SET SQL_MODE=@OLD_SQL_MODE;
648 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
649 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

APÊNDICE C – Código fonte Tweet Context Collector

Listing C.1 – code/MainActivity.java

```
1 package com.example.karran.myapplication;
2
3 import android.content.BroadcastReceiver;
4 import android.content.ContentValues;
5 import android.content.Context;
6 import android.content.Intent;
7 import android.content.IntentFilter;
8 import android.content.pm.PackageManager;
9 import android.database.Cursor;
10 import android.net.Uri;
11 import android.os.AsyncTask;
12 import android.os.Handler;
13 import android.support.v4.content.ContextCompat;
14 import android.support.v7.app.AppCompatActivity;
15 import android.os.Bundle;
16 import android.util.Log;
17 import android.view.View;
18 import android.widget.TextView;
19
20 import com.aware.Accelerometer;
21 import com.aware.Applications;
22 import com.aware.Aware;
23 import com.aware.Aware_Preferences;
24 import com.aware.Barometer;
25 import com.aware.Gyroscope;
26 import com.aware.Light;
27 import com.aware.Magnetometer;
28 import com.aware.Rotation;
29 import com.aware.Temperature;
30 import com.aware.plugin.google.activity_recognition.
    Google_AR_Provider;
31 import com.aware.plugin.google.fused_location.Provider;
32 import com.aware.providers.Accelerometer_Provider;
33 import com.aware.providers.Applications_Provider;
34 import com.aware.providers.Barometer_Provider;
35 import com.aware.providers.Gravity_Provider;
36 import com.aware.providers.Gyroscope_Provider;
37 import com.aware.providers.Light_Provider;
```

```
38 import com.aware.providers.Locations_Provider;
39 import com.aware.providers.Magnetometer_Provider;
40 import com.aware.providers.Rotation_Provider;
41 import com.aware.providers.Temperature_Provider;
42 import com.aware.ui.PermissionsHandler;
43
44 import java.io.BufferedInputStream;
45 import java.io.BufferedReader;
46 import java.io.IOException;
47 import java.io.InputStream;
48 import java.io.InputStreamReader;
49 import java.io.OutputStream;
50 import java.net.HttpURLConnection;
51 import java.net.URL;
52 import java.util.ArrayList;
53 import java.util.Timer;
54 import java.util.TimerTask;
55
56 import javax.net.ssl.HttpURLConnection;
57
58 import static com.aware.Aware.stopAccelerometer;
59 import static com.aware.Aware.stopBarometer;
60 import static com.aware.Aware.stopGravity;
61 import static com.aware.Aware.stopGyroscope;
62 import static com.aware.Aware.stopLight;
63 import static com.aware.Aware.stopMagnetometer;
64 import static com.aware.Aware.stopRotation;
65 import static com.aware.Aware.stopTemperature;
66
67 public class MainActivity extends AppCompatActivity {
68     private StatusListener ssl = new StatusListener();
69     private ArrayList<String> REQUIRED_PERMISSIONS = new ArrayList
70         <>();
71     private Timer timer = new Timer();
72     private ArrayList<String> sensorsList= new ArrayList();
73     private ArrayList<Uri> uriList= new ArrayList();
74     private int actualSensorId = 0;
75     private boolean makeSync = false;
76     private boolean onProcess = false;
77
78     class TpauseSensors extends TimerTask {
79         public void run() {
80             setSensorsEnabled(false);
81             if (actualSensorId==0) {
82                 if (!makeSync) {
83                     sendLowFrequency();
84                     timer.cancel();
```

```
84         return;
85     }
86     Aware.startAccelerometer(getApplicationContext());
87     actualSensorId++;
88 } else if (actualSensorId==1){
89     Aware.startBarometer(getApplicationContext());
90     actualSensorId++;
91 } else if (actualSensorId==2){
92     Aware.startGravity(getApplicationContext());
93     actualSensorId++;
94 } else if (actualSensorId==3){
95     Aware.startGyroscope(getApplicationContext());
96     actualSensorId++;
97 }else if (actualSensorId==4){
98     Aware.startLight(getApplicationContext());
99     actualSensorId++;
100 }else if (actualSensorId==5){
101     Aware.startMagnetometer(getApplicationContext());
102     actualSensorId++;
103 }else if (actualSensorId==6){
104     Aware.startRotation(getApplicationContext());
105     actualSensorId++;
106 }else if (actualSensorId==7) {
107     Aware.startTemperature(getApplicationContext());
108     actualSensorId++;
109 }else {
110     makeSync=false;
111     actualSensorId=0;
112 }
113 }
114 }
115
116 @Override
117 protected void onCreate(Bundle savedInstanceState) {
118     super.onCreate(savedInstanceState);
119     setContentView(R.layout.activity_main);
120     Intent aware = new Intent(this, Aware.class);
121     startService(aware);
122     boolean permissions_ok = true;
123     for (String p : REQUIRED_PERMISSIONS) {
124         if (ContextCompat.checkSelfPermission(this, p) !=
125             PackageManager.PERMISSION_GRANTED) {
126             permissions_ok = false;
127             break;
128         }
129     }
130     if (!permissions_ok) {
```

```
130         Intent permissions = new Intent(this,
131             PermissionsHandler.class);
132         permissions.putExtra(PermissionsHandler.
133             EXTRA_REQUIRED_PERMISSIONS, REQUIRED_PERMISSIONS);
134         permissions.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
135
136         startActivity(permissions);
137         finish();
138     } else {
139         Applications.isAccessibilityServiceActive(
140             getApplicationContext());
141     }
142     Applications.isAccessibilityServiceActive(
143         getApplicationContext());
144     Aware.setSetting(getApplicationContext(),
145         Aware_Preferences.STATUS_NOTIFICATIONS, true);
146     IntentFilter application_filter = new IntentFilter();
147     application_filter.addAction(Applications.
148         ACTION_AWARE_APPLICATIONS_NOTIFICATIONS);
149     NotificationObserver so = new NotificationObserver(new
150         Handler(), this, this);
151     getContentResolver().registerContentObserver(
152         Applications_Provider.Applications_Notifications.
153         CONTENT_URI, true, so);
154     Aware.joinStudy(getApplicationContext(), "https://api.
155         awareframework.com/index.php/webservice/index/789/
156         kyX1IoIPqcJI");
157     sensorsList.add(Aware_Preferences.STATUS_ACCELEROMETER);
158     sensorsList.add(Aware_Preferences.STATUS_APPLICATIONS);
159     sensorsList.add(Aware_Preferences.STATUS_BAROMETER);
160     sensorsList.add(Aware_Preferences.STATUS_GRAVITY);
161     sensorsList.add(Aware_Preferences.STATUS_GYROSCOPE);
162     sensorsList.add(Aware_Preferences.STATUS_LIGHT);
163     sensorsList.add(Aware_Preferences.STATUS_LOCATION_GPS);
164     sensorsList.add(Aware_Preferences.STATUS_LOCATION_NETWORK)
165         ;
166     sensorsList.add(Aware_Preferences.STATUS_MAGNETOMETER);
167     sensorsList.add(Aware_Preferences.STATUS_ROTATION);
168     sensorsList.add(Aware_Preferences.STATUS_TEMPERATURE);
169     uriList.add(Accelerometer_Provider.Accelerometer_Data.
170         CONTENT_URI);
171     uriList.add(Barometer_Provider.Barometer_Data.CONTENT_URI)
172         ;
173     uriList.add(Gravity_Provider.Gravity_Data.CONTENT_URI);
174     uriList.add(Gyroscope_Provider.Gyroscope_Data.CONTENT_URI)
175         ;
176     uriList.add(Light_Provider.Light_Data.CONTENT_URI);
```



```
162         uriList.add(Locations_Provider.Locations_Data.CONTENT_URI)
163         ;
164         uriList.add(Magnetometer_Provider.Magnetometer_Data.
165             CONTENT_URI);
166         uriList.add(Rotation_Provider.Rotation_Data.CONTENT_URI);
167         uriList.add(Temperature_Provider.Temperature_Data.
168             CONTENT_URI);
169         Aware.setSetting(this, Aware_Preferences.
170             FREQUENCY_ACCELEROMETER, 0);
171         Aware.setSetting(this, Aware_Preferences.
172             FREQUENCY_BAROMETER, 0);
173         Aware.setSetting(this, Aware_Preferences.FREQUENCY_GRAVITY
174             , 0);
175         Aware.setSetting(this, Aware_Preferences.
176             FREQUENCY_GYROSCOPE, 0);
177         Aware.setSetting(this, Aware_Preferences.FREQUENCY_LIGHT,
178             0);
179         Aware.setSetting(this, Aware_Preferences.
180             FREQUENCY_MAGNETOMETER, 0);
181         Aware.setSetting(this, Aware_Preferences.
182             FREQUENCY_ROTATION, 0);
183         Aware.setSetting(this, Aware_Preferences.
184             FREQUENCY_TEMPERATURE, 0);
185         Aware.setSetting(this, Aware_Preferences.
186             FREQUENCY_CLEAN_OLD_DATA, 4);
187         Aware.setSetting(this, Aware_Preferences.
188             STATUS_ACCELEROMETER, false);
189         Aware.setSetting(this, Aware_Preferences.STATUS_BAROMETER,
190             false);
191         Aware.setSetting(this, Aware_Preferences.STATUS_GRAVITY,
192             false);
193         Aware.setSetting(this, Aware_Preferences.STATUS_GYROSCOPE,
194             false);
195         Aware.setSetting(this, Aware_Preferences.STATUS_LIGHT,
196             false);
197         Aware.setSetting(this, Aware_Preferences.
198             STATUS_LOCATION_GPS, false);
199         Aware.setSetting(this, Aware_Preferences.
200             STATUS_LOCATION_NETWORK, false);
201         Aware.setSetting(this, Aware_Preferences.
202             STATUS_MAGNETOMETER, false);
203         Aware.setSetting(this, Aware_Preferences.STATUS_ROTATION,
204             false);
205         Aware.setSetting(this, Aware_Preferences.
206             STATUS_TEMPERATURE, false);
207         Aware.setSetting(this, Aware_Preferences.STATUS_WIFI,
208             false);
```

```
186     Aware.setSetting(getApplicationContext(),
187         Aware_Preferences.STATUS_APPLICATIONS, true);
188     IntentFilter application_filter2 = new IntentFilter();
189     application_filter2.addAction(Applications.
190         ACTION_AWARE_APPLICATIONS_FOREGROUND);
191     TextView txt = (TextView) findViewById(R.id.txt_aplicativo
192         );
193     txt.setText("UUID: " + Aware.getSetting(this,
194         Aware_Preferences.DEVICE_ID));
195     Aware.setSetting(this, Aware_Preferences.
196         STATUS_LOCATION_NETWORK, true);
197     Aware.setSetting(this, Aware_Preferences.
198         STATUS_LOCATION_GPS, true);
199     IntentFilter broadcastFilter = new IntentFilter();
200     broadcastFilter.addAction(Accelerometer.
201         ACTION_AWARE_ACCELEROMETER);
202     broadcastFilter.addAction(Barometer.ACTION_AWARE_BAROMETER
203         );
204     broadcastFilter.addAction(com.aware.Gravity.
205         ACTION_AWARE_GRAVITY);
206     broadcastFilter.addAction(Gyroscope.ACTION_AWARE_GYROSCOPE
207         );
208     broadcastFilter.addAction(Magnetometer.
209         ACTION_AWARE_MAGNETOMETER);
210     broadcastFilter.addAction(Rotation.ACTION_AWARE_ROTATION);
211     broadcastFilter.addAction(Temperature.
212         ACTION_AWARE_TEMPERATURE);
213     broadcastFilter.addAction(Light.ACTION_AWARE_LIGHT);
214     registerReceiver(ssl, broadcastFilter);
215     Aware.startLocations(getApplicationContext());
216     Aware.startPlugin(this, "com.aware.plugin.google.
217         activity_recognition.lib");
218     Aware.startPlugin(this, "com.aware.plugin.google.
219         fused_location.lib");
220 }
221
222 public void setSensorsEnabled(boolean _enable){
223     Aware.stopAccelerometer(this);
224     Aware.stopBarometer(this);
225     Aware.stopGravity(this);
226     Aware.stopGyroscope(this);
227     Aware.stopLight(this);
228     Aware.stopMagnetometer(this);
229     Aware.stopRotation(this);
230     Aware.stopTemperature(this);
231     return;
232 }
```

```
219
220     public void onClick(View v){
221         if (!onProcess) {
222             onProcess=true;
223             actualSensorId = 0;
224             makeSync = true;
225             timer = new Timer();
226             TimerTask pauseSensors = new TpauseSensors();
227             timer.scheduleAtFixedRate(pauseSensors, 0, 1000);
228         }
229     }
230
231     private void sendLowFrequency() {
232         Cursor c = getContentResolver().query(Locations_Provider.
                Locations_Data.CONTENT_URI, null, null, null,
                Locations_Provider.Locations_Data.TIMESTAMP + " DESC
                LIMIT 1");
233         if (c.moveToFirst()) {
234             String query = "";
235             Uri.Builder builder = new Uri.Builder();
236             builder = new Uri.Builder().appendQueryParameter("_timestamp", c.getString(c.getColumnIndex("
                timestamp")))
237                 .appendQueryParameter("_device", Aware.
                getSetting(this, Aware_Preferences.
                DEVICE_ID))
238                 .appendQueryParameter("_sensor", "location")
239                 .appendQueryParameter("_accuracy", c.getString
                (c.getColumnIndex("accuracy")))
240                 .appendQueryParameter("_double_latitude", c.
                getString(c.getColumnIndex("double_latitude
                ")))
241                 .appendQueryParameter("_double_longitude", c.
                getString(c.getColumnIndex("
                double_longitude")))
242                 .appendQueryParameter("_double_speed", c.
                getString(c.getColumnIndex("double_speed"))
                )
243                 .appendQueryParameter("_double_altitude", c.
                getString(c.getColumnIndex("double_altitude
                ")))
244                 .appendQueryParameter("_provider", c.getString
                (c.getColumnIndex("provider")))
245                 .appendQueryParameter("_double_bearing", c.
                getString(c.getColumnIndex("double_bearing"
                )))
                );
246             query = builder.build().getEncodedQuery();
```

```
247         new AsyncLoad(query).execute();
248     }
249     c = getContentResolver().query(Google_AR_Provider.
        Google_Activity_Recognition_Data.CONTENT_URI, null,
        null, null, Google_AR_Provider.
        Google_Activity_Recognition_Data.TIMESTAMP + " DESC
        LIMIT 1");
250     if (c.moveToFirst()) {
251         String query = "";
252         Uri.Builder builder = new Uri.Builder();
253         builder = new Uri.Builder().appendQueryParameter("
            _timestamp", String.valueOf(c.getDouble(c.
            getColumnIndex("timestamp"))))
254             .appendQueryParameter("_device", Aware.
                getSetting(this, Aware_Preferences.
                DEVICE_ID))
255             .appendQueryParameter("_sensor", "activity")
256             .appendQueryParameter("_activity_name", c.
                getString(c.getColumnIndex("activity_name")
                ))
257             .appendQueryParameter("_activity_type", c.
                getString(c.getColumnIndex("activity_type")
                ))
258             .appendQueryParameter("_confidence", c.
                getString(c.getColumnIndex("confidence")))
259             .appendQueryParameter("_activities", c.
                getString(c.getColumnIndex("activities")));
260         query = builder.build().getEncodedQuery();
261         new AsyncLoad(query).execute();
262     }
263     c = getContentResolver().query(Provider.Geofences_Data.
        CONTENT_URI, null, null, null, Provider.Geofences_Data.
        TIMESTAMP + " DESC LIMIT 1");
264     if (c.moveToFirst()) {
265         String query = "";
266         Uri.Builder builder = new Uri.Builder();
267         builder = new Uri.Builder().appendQueryParameter("
            _timestamp", c.getString(c.getColumnIndex("
            timestamp")))
268             .appendQueryParameter("_device", Aware.
                getSetting(this, Aware_Preferences.
                DEVICE_ID))
269             .appendQueryParameter("_sensor", "geofence")
270             .appendQueryParameter("_geofence_label", c.
                getString(c.getColumnIndex("geofence_label"
                )))
271             .appendQueryParameter("_double_latitude", c.
```

```
                getString(c.getColumnIndex("double_latitude
272                .appendQueryParameter("_double_longitude", c.
                getString(c.getColumnIndex("
                double_longitude")))
273                .appendQueryParameter("_double_distance", c.
                getString(c.getColumnIndex("double_distance
                ")))
274                .appendQueryParameter("_status", c.getString(c
                .getColumnIndex("status")));
275                query = builder.build().getEncodedQuery();
276                new AsyncLoad(query).execute();
277            }
278            onProcess=false;
279            deleteAllButMostRecent();
280        }
281
282        public class StatusListener extends BroadcastReceiver {
283            public void onReceive(Context c, Intent intent) {
284                String query = "";
285                Uri.Builder builder = new Uri.Builder();
286                ContentValues latest;
287                Log.d("MAINACTIVITY", intent.getAction());
288                switch (intent.getAction()){
289                    case "ACTION_AWARE_ACCELEROMETER":
290                        stopAccelerometer(getApplicationContext());
291                        latest = (ContentValues) intent.
                            getParcelableExtra(Accelerometer.EXTRA_DATA
                            );
292                        builder = new Uri.Builder().
                            appendQueryParameter("_timestamp", latest.
                            getAsString("timestamp"))
293                            .appendQueryParameter("_device", Aware
                            .getSetting(getApplicationContext()
                            , Aware_Preferences.DEVICE_ID))
294                            .appendQueryParameter("_sensor", "
                            accelerometer")
295                            .appendQueryParameter("_accuracy",
                            latest.getAsString("accuracy"))
296                            .appendQueryParameter("
                            _double_values_0", latest.
                            getAsString("double_values_0"))
297                            .appendQueryParameter("
                            _double_values_1", latest.
                            getAsString("double_values_1"))
298                            .appendQueryParameter("
                            _double_values_2", latest.
```

```
                getAsString("double_values_2"));
299         query = builder.build().getEncodedQuery();
300         new AsyncLoad(query).execute();
301         break;
302     case "ACTION_AWARE_BAROMETER":
303         stopBarometer(getApplicationContext());
304         latest = (ContentValues) intent.
            getParcelableExtra(Barometer.EXTRA_DATA);
305         builder = new Uri.Builder().
            appendQueryParameter("_timestamp", latest.
                getAsString("timestamp"))
306             .appendQueryParameter("_device", Aware
                .getSetting(getApplicationContext()
                    , Aware_Preferences.DEVICE_ID))
307             .appendQueryParameter("_sensor", "
                barometer")
308             .appendQueryParameter("_accuracy",
                latest.getAsString("accuracy"))
309             .appendQueryParameter("
                _double_values_0", latest.
                getAsString("double_values_0"));
310         query = builder.build().getEncodedQuery();
311         new AsyncLoad(query).execute();
312         break;
313     case "ACTION_AWARE_GRAVITY":
314         stopGravity(getApplicationContext());
315         latest = (ContentValues) intent.
            getParcelableExtra(com.aware.Gravity.
                EXTRA_DATA);
316         builder = new Uri.Builder().
            appendQueryParameter("_timestamp", latest.
                getAsString("timestamp"))
317             .appendQueryParameter("_device", Aware
                .getSetting(getApplicationContext()
                    , Aware_Preferences.DEVICE_ID))
318             .appendQueryParameter("_sensor", "
                gravity")
319             .appendQueryParameter("_accuracy",
                latest.getAsString("accuracy"))
320             .appendQueryParameter("
                _double_values_0", latest.
                getAsString("double_values_0"))
321             .appendQueryParameter("
                _double_values_1", latest.
                getAsString("double_values_1"))
322             .appendQueryParameter("
                _double_values_2", latest.
```

```
323         getAsString("double_values_2"));
324     query = builder.build().getEncodedQuery();
325     new AsyncLoad(query).execute();
326     break;
327 case "ACTION_AWARE_GYROSCOPE":
328     stopGyroscope(getApplicationContext());
329     latest = (ContentValues) intent.
330         getParcelableExtra(Gyroscope.EXTRA_DATA);
331     builder = new Uri.Builder().
332         appendQueryParameter("_timestamp", latest.
333             getAsString("timestamp"))
334             .appendQueryParameter("_device", Aware
335                 .getSetting(getApplicationContext()
336                     , Aware_Preferences.DEVICE_ID))
337             .appendQueryParameter("_sensor", "
338                 gyroscope")
339             .appendQueryParameter("_accuracy",
340                 latest.getAsString("accuracy"))
341             .appendQueryParameter("_double_values_0", latest.
342                 getAsString("double_values_0"))
343             .appendQueryParameter("_double_values_1", latest.
344                 getAsString("double_values_1"))
345             .appendQueryParameter("_double_values_2", latest.
346                 getAsString("double_values_2"));
347     query = builder.build().getEncodedQuery();
348     new AsyncLoad(query).execute();
349     break;
350 case "ACTION_AWARE_MAGNETOMETER":
351     stopMagnetometer(getApplicationContext());
352     latest = (ContentValues) intent.
353         getParcelableExtra(Magnetometer.EXTRA_DATA)
354         ;
355     builder = new Uri.Builder().
356         appendQueryParameter("_timestamp", latest.
357             getAsString("timestamp"))
358             .appendQueryParameter("_device", Aware
359                 .getSetting(getApplicationContext()
360                     , Aware_Preferences.DEVICE_ID))
361             .appendQueryParameter("_sensor", "
362                 magnetometer")
363             .appendQueryParameter("_accuracy",
364                 latest.getAsString("accuracy"))
365             .appendQueryParameter("_double_values_0", latest.
```

```
347         getString("double_values_0"))
        .appendQueryParameter("
348             _double_values_1", latest.
            getString("double_values_1"))
        .appendQueryParameter("
            _double_values_2", latest.
            getString("double_values_2"));
349     query = builder.build().getEncodedQuery();
350     new AsyncLoad(query).execute();
351     break;
352
353     case "ACTION_AWARE_TEMPERATURE":
354         stopTemperature(getApplicationContext());
355         latest = (ContentValues) intent.
            getParcelableExtra(Temperature.EXTRA_DATA);
356         builder = new Uri.Builder().
            appendQueryParameter("_timestamp", latest.
            getString("timestamp"))
357             .appendQueryParameter("_device", Aware
                .getSetting(getApplicationContext()
                    , Aware_Preferences.DEVICE_ID))
358             .appendQueryParameter("_sensor", "
                temperature")
359             .appendQueryParameter("_accuracy",
                latest.getString("accuracy"))
360             .appendQueryParameter("
                _temperature_celsius", latest.
                getString("temperature_celsius"))
            ;
361         query = builder.build().getEncodedQuery();
362         new AsyncLoad(query).execute();
363         break;
364     case "ACTION_AWARE_ROTATION":
365         stopRotation(getApplicationContext());
366         latest = (ContentValues) intent.
            getParcelableExtra(Rotation.EXTRA_DATA);
367         builder = new Uri.Builder().
            appendQueryParameter("_timestamp", latest.
            getString("timestamp"))
368             .appendQueryParameter("_device", Aware
                .getSetting(getApplicationContext()
                    , Aware_Preferences.DEVICE_ID))
369             .appendQueryParameter("_sensor", "
                rotation")
370             .appendQueryParameter("_accuracy",
                latest.getString("accuracy"))
371             .appendQueryParameter("
            
```



```
372         _double_values_0", latest.  
            getAsString("double_values_0"))  
        .appendQueryParameter("  
373         _double_values_1", latest.  
            getAsString("double_values_1"))  
        .appendQueryParameter("  
374         _double_values_2", latest.  
            getAsString("double_values_2"))  
        .appendQueryParameter("  
            _double_values_3", latest.  
            getAsString("double_values_3"));  
375     query = builder.build().getEncodedQuery();  
376     new AsyncLoad(query).execute();  
377     break;  
378     case "ACTION_AWARE_LIGHT":  
379         stopLight(getApplicationContext());  
380         latest = (ContentValues) intent.  
            getParcelableExtra(Light.EXTRA_DATA);  
381         builder = new Uri.Builder().  
            appendQueryParameter("_timestamp", latest.  
                getAsString("timestamp"))  
382             .appendQueryParameter("_device", Aware  
                .getSetting(getApplicationContext()  
                    , Aware_Preferences.DEVICE_ID))  
383             .appendQueryParameter("_sensor", "  
                light")  
384             .appendQueryParameter("_accuracy",  
                latest.getAsString("accuracy"))  
385             .appendQueryParameter("  
                _double_light_lux", latest.  
                getAsString("double_light_lux"));  
386         query = builder.build().getEncodedQuery();  
387         new AsyncLoad(query).execute();  
388         break;  
389     default:  
390         break;  
391     }  
392 }  
393 }  
394  
395 public void deleteAllButMostRecentFrom(Uri _uri){  
396     getContentResolver().delete(_uri, null, null);  
397 }  
398  
399 public void deleteAllButMostRecent(){  
400     deleteAllButMostRecentFrom(Accelerometer_Provider.  
        Accelerometer_Data.CONTENT_URI);
```

```
401         deleteAllButMostRecentFrom(Gravity_Provider.Gravity_Data.
           CONTENT_URI);
402         deleteAllButMostRecentFrom(Light_Provider.Light_Data.
           CONTENT_URI);
403         deleteAllButMostRecentFrom(Locations_Provider.
           Locations_Data.CONTENT_URI);
404         deleteAllButMostRecentFrom(Magnetometer_Provider.
           Magnetometer_Data.CONTENT_URI);
405         deleteAllButMostRecentFrom(Rotation_Provider.Rotation_Data
           .CONTENT_URI);
406         deleteAllButMostRecentFrom(Temperature_Provider.
           Temperature_Data.CONTENT_URI);
407         deleteAllButMostRecentFrom(Locations_Provider.
           Locations_Data.CONTENT_URI);
408         deleteAllButMostRecentFrom(Google_AR_Provider.
           Google_Activity_Recognition_Data.CONTENT_URI);
409         deleteAllButMostRecentFrom(Provider.Geofences_Data.
           CONTENT_URI);
410     }
411
412     public void notificationObserved(){
413         Cursor notification = getContentResolver().query(
           Applications_Provider.Applications_Notifications.
           CONTENT_URI, null, "application_name = 'Twitter'", null
           , "TIMESTAMP DESC");
414         if (notification.moveToFirst()) {
415             getContentResolver().delete(Applications_Provider.
           Applications_Notifications.CONTENT_URI, "
           application_name='Twitter'", null);
416             onProcess=true;
417             timer = new Timer();
418             actualSensorId=0;
419             makeSync=true;
420             TimerTask pauseSensors = new TpauseSensors();
421             timer.scheduleAtFixedRate(pauseSensors, 0, 1000);
422         }
423         notification.close();
424     }
425
426     class AsyncLoad extends AsyncTask<Void,Void,Void>
427     {
428         InputStream inputStream;
429         HttpURLConnection urlConnection;
430         byte[] outputBytes;
431         String query;
432         String responseData;
433         public AsyncLoad(String query) {
```

```
434         this.query = query;
435     }
436     @Override
437     protected Void doInBackground(Void... params) {
438         try {
439             URL url = new URL("http://www.pbeedifica.com.
440                 br/tcc.php");
441             urlConnection = (HttpURLConnection) url.
442                 openConnection();
443             outputBytes = query.getBytes("UTF-8");
444             urlConnection.setRequestMethod("POST");
445             urlConnection.connect();
446             OutputStream os = urlConnection.
447                 getOutputStream();
448             os.write(outputBytes);
449             os.close();
450             int statusCode = urlConnection.getResponseCode
451                 ();
452             if (statusCode == HttpURLConnection.HTTP_OK)
453                 {
454                     inputStream = new BufferedInputStream(
455                         urlConnection.getInputStream());
456                     responseData = convertStreamToString(
457                         inputStream);
458                 } else {
459                     responseData = null;
460                 }
461             } catch (Exception e) {
462                 e.printStackTrace();
463             }
464             return null;
465         }
466     }
467
468     public static String convertStreamToString(InputStream is) {
469         BufferedReader reader = new BufferedReader(new
470             InputStreamReader(is));
471         StringBuilder sb = new StringBuilder();
472         String line = null;
473         try {
474             while ((line = reader.readLine()) != null) {
475                 sb.append((line + "\n"));
476             }
477         } catch (IOException e) {
478             e.printStackTrace();
479         } finally {
480             try {
```

```
473         is.close();
474     } catch (IOException e) {
475         e.printStackTrace();
476     }
477 }
478 return sb.toString();
479 }
480 }
```

Listing C.2 – code/NotificationObserver.java

```
1 package com.example.karran.myapplication;
2
3 import android.content.Context;
4 import android.database.ContentObserver;
5 import android.os.Handler;
6 import android.database.Cursor;
7 import android.util.Log;
8 import android.widget.TextView;
9
10 import com.aware.providers.Applications_Provider;
11
12 public class NotificationObserver extends ContentObserver {
13     Context context;
14     MainActivity act;
15     public NotificationObserver(Handler handler, Context context,
16         MainActivity _act) {
17         super(handler);
18         this.context = context;
19         this.act = _act;
20     }
21
22     @Override
23     public void onChange(boolean selfChange) {
24         act.notificationObserved();
25     }
26 }
```