Free and Open Source Software for Geospatial (FOSS4G) Conference Proceedings

Volume 13 *Nottingham, UK* Article 6

2013

Taarifa - Improving Public Service Provision in the Developing World Through a Crowd-sourced Location Based Reporting Application

Mark Iliffe
University of Nottingham (UK)

Giuseppe Sollazzo
St. George's University of London (UK)

Jeremy Morley
University of Nottingham (UK)

Robert Houghton
University of Nottingham (UK)

Taarifa

Improving Public Service Provision in the Developing World Through a Crowd-sourced Location Based Reporting Application

by Mark Iliffe[1]*, Giuseppe Sollazzo*[2]*, Jeremy Morley*[1]*, and Robert Houghton*[1]

*1: University of Nottingham (UK); 2: St. George's University of London (UK).*psxmi@nottingham.ac.uk

Abstract

Public service provision in the developing world is challenged by a lack of coherence and consistency in the amount of resources local authorities have in their endowment. Especially where non-planned urban settlements (e.g. slums) are present, the frequent and constant change of the urban environment poses big challenges to the effective delivery of services. In this paper we report on our experiences with Taarifa: a location-based application built through community development that allows community reporting and managing of local issues.

Keywords: Location-based application, community development, crowd-sourcing.

Introduction

The availability of geographic data in the developing world is improving with the advent of community mapping projects like Map Kibera [1, 2] and through organisations like Humanitarian Open Street Map Team (H.O.T). Previously a large barrier for NGOs, governments and business in providing services in developing world, the lack of governmental and non-governmental data is becoming an issue of the past and with this barrier rapidly dissolving further questions are arising, such as: now we have the data, what do we do next?

The Taarifa project aims to address this question, with respect to the monitoring of public service provision. Taarifa as a software platform allows for the community reporting of problems, from health to waste issues, through a mobile phone interface using SMS or a HTML5 client. Once reports are collected they are entered into a workflow allowing those in charge of providing services to monitor, triage and act upon reports.

Taarifa is currently unique in this field from its initial design, inception and deployment. It was originally conceived at the Random Hacks of Kindness (RHOK) London Water Hackathon. A Hackathon is an organised meeting of developers who team up to code on a specific topic or to address a specific problem. Hackathon are very popular in the developers community. Both private and public organisations often set up hackathons to get some fresh hands working on certain issues they are facing. During the RHOK hackathon a group of core developers 'hacked' a solution in 48 hours. After continued development and design Taarifa was first deployed in cooperation with the Ugandan Ministry of Local Government in March 2012, followed by a deployment with the Zimbabwean Government in April 2012 facilitated by the World Bank.

In this paper we present a narrative and case study of the Taarifa project from its inception, design, refinement and deployment. We also discuss the future directions Taarifa might take both as a community software project and as an organisation. The global aim is to facilitate a discussion on how crowd-sourced geospatial data and open source platforms can combine to improve public and private service delivery in developing nations.

Related Work

Most related work considering the emergent phenomena of crisis mapping are case studies of specific crises, the Haiti Earthquake [3], terrorist attacks [4], methodologies for crisis situation triage [5]. Though, in context not all methodologies of crisis relief are wholly focused on external response as [6] demonstrates. These instances of citizens generating reports fall under the banner of crowdsourcing. Here [7] specifically looks at providing situational awareness; how a visual 'group map' of all the reports is useful as errors can be made. The crowd sourcing of information has inherent dangers of trusting the information supplied, as [8] demonstrates with respect to potential crisis situations.

Currently there are two themes missing from the literature; A study of how the reports are being used to aid decisions and an understanding of areas in a constant state of crisis. These areas, like slums and informal developments do not have an event like an earthquake or a tsunami to illustrate the plight. This is combined by [9] review of the Map Kibera project.

Almost anecdotal evidence [10, 11] exists to how the crowd sourced data is used, but nothing to the experiences of using it, within the field of crisis mapping. However, numerous papers cover the experiences of the crowd and their validation in citizen science, like ecology [12]. A possible output and definite gap in the research would be an ethnographic study of when a crisis occurs, observing and reporting the 'value chain' and how the data is used. A noteworthy omission from this section of the review is ICT4D field of study. While this is a rapidly developing field the research seems to be based more on the social science fringe compared to the more software/algorithm development side, though the gap is being bridged.

The effect of social media, indicates that on a social level it aids the transition to recovery through blogs [13, 14] and as a general community platform to generate maps [10]

Taarifa's inception at the RHOK Hackathon

The London Random Hacks of Kindness Hackathon (RHoK) occurred on Friday 21st of October, 2011, lasting for 48 hours in the facilities of University College London. Randomly assembled groups of coders with interests on humanitarian subjects and matter experts joined forces to work together with the aim of producing technology demonstrators and designs to solve problems related to water.

Why A Hackathon?

The underlying idea of any hackathon, and in particular of the RHOK Hackathon, is that by co-locating intelligent, innovative and driven computer software developers and field experts, facilitating a fast-paced production session lasting no more than a couple of days, can lead to worthwhile innovation. One condition is that all software produced has to be open source, and this naturally offers opportunity of further development by different teams in the future. There are many kinds of hackathons: from those destined to the very young, to corporate-run hackathons in which participants are selected and paid a daily fee. Many question have arisen about the effectiveness of hackathons in producing something really useful, especially generalist hacking events. However, we argue that dedicated hackathons like the RHOK are a practical and effective way of developing products, mostly due to their very narrow target. They can be seen as "sprints" sessions as formalised by the Agile project management theory [15].

A story telling of the event

The event started with an introduction by the organisers framing the exercise with problem statements and presentations by experts wanting to solve different problems. Some of these people spoke in person at the event, while others via teleconference. The problems presented all came form the RHoK website and ranged from water trading platforms to public service infrastructures and community mapping.

Figure 1: London WaterHackathon 2011

Taarifa started to take shape of a platform to support citizen interaction around public services for lesser developed countries.

A team was assembled and collectively started to set up whiteboards and tools to construct the intended processes and a design specification for workflows through the potential system. The discussion focussed on how triaging of reports would work. At the time Taarifa's intended customer was meant to come from a ministry level, potentially working around sanitation or waste issues.

Aiming for rapid development of the platform we decided to fork the Ushahidi platform (Okolloh 2009). Ushahidi is a platform/CMS designed for the crowdsourced reporting of issues, its inception was due to the Kenyan election crisis of 2008, since has been used to report conflicts like civil wars but also the recent 'occupy' movements. Technologically it's foundations are built in PHP using the Kohana framework. Here we hit our first issue; none of our developers had worked with Kohana before. We split into two groups, one figuring out Kohana, the other designing workflow.

It became apparent that experience in the right tools was needed with some developers wishing to contribute. However, some were not able to operate at the level as some developers, which is typical of hackathons where a large team works together. As this progressed, these developers filled other roles, testing and aiding the main thrust instead of contributing code. However, their contribution was as valuable in real terms as the code generated. Installing Ushahidi also proved problematic. Issues with mod_rewrite and other PHP extensions were experienced, but were eventually resolved. These potentially could have been avoided through enhanced documentation. Equipment at the hackathon was problematic: the team didn't have access to hosted server, hence one of the developers' personal servers was used.

Once the workflow was sketched out we presented back to the other team of developers. They had conducted a study into the Ushahidi plugin ecosystem. Collectively we integrated the 'actionable' and 'simple groups' plugins. Actionable was adapted to 'action' reports, and place them in the triage system. Simple groups was used to curate a team of 'fixers'. Fixers was used generically as the people fixing the reported problems, however the dynamic of how this would be fully implemented wasn't considered at this stage.

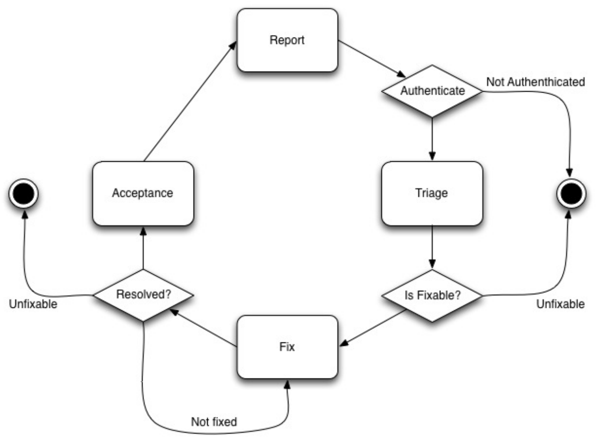


Figure 2: Workflow Of Reports Through The Taarifa System

Inheriting the interface from Ushahidi meant the load on the developers was focused towards the back-end with tasks and problems being received and triaged. The workflow started to come together, based on the idea of community reports being verified, then put on triage, assigned to a team of fixers and finally reaching conclusion or, if not appropriate, directed to dispute resolution. The visual user interface, organised in tabs to accommodate these functions, was integrated into the system. As reports were able to be triaged we focused on expanding the reporting mechanism. Ushahidi supports reporting through a web-based form, twitter and through its mobile applications (iOS, Android, Java and Windows Mobile 6). It can interface with SMS gateways like FrontlineSMS. The team intended to use SMS due to the ubiquitous nature of feature phones in Africa that realistically can only use SMS as a form of reporting. Using SMS presented problems of geolocating the messages.

The OGC [16] standard on GeoSMS was unfortunately unavailable at the time, it is possible for the mobile phone networks to triangulate the position of the sender and supply a latitude and longitude however this isn't practical over a 48 hour hackathon notwithstanding the ethical and privacy concerns.

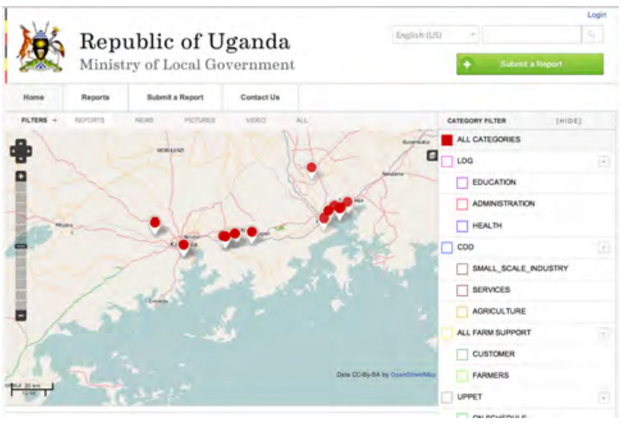
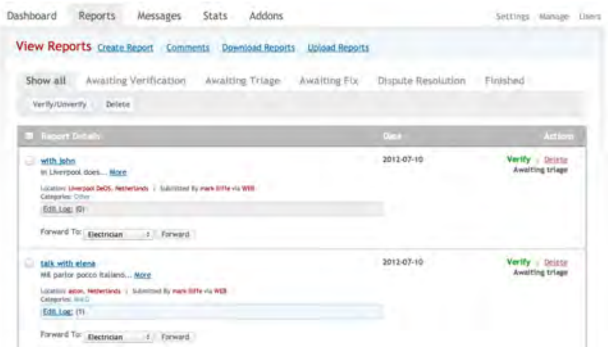


Figure 3: Taarifa Interface



with John
in Liverpool does... More
Location: Liverpool DeOS, Netherlands | Submitted By mark iliffe via WEB
Categories: Other
Edit Log: (0)
2012-07-10
Awaiting triage

talk with elena
Mi parlor pocco italiano... More
Location: aeon, Netherlands | Submitted By mark iliffe via WEB
Categories: Wel.D
Edit Log: (1)
2012-07-10
Awaiting triage

Figure 4: Taarifa Workflow Management

In response to this issue a 100m^2 grid was created under a custom coordinate reference system having a 10 digit reference for each grid square. Then a reporter with a SMS capable phone would input the number with a hash (#) then found through a regular expression in the submitted message. Obviously questions remain when implementing this on a large scale namely ensuring local people know what their code is and creating a reference system that conforms to the human geography not just the physical. This was accomplished on the first day of the hackathon, however we worked through the night, resting for four hours. A global team of friends carried on completing an SMS gateway. What remained at the hackathon around bug-fixing, tidying up code starting documentation and choosing a name. One team member searched the word “Reporting” into a translation software for the Swahili language. ’Taarifa’ was the result.

Discussion about the results of the hackathon

The Taarifa group at the hackathon was fortunate to be successful and be voted winners of the London hackathon. It was decided that the project was an interesting effort to address real problem, and should be kept alive. The team had worked and synchronised well. Administratively an online mailing list was created, communication through instant messaging and logos and branding. The team assessed that the integration of the mobile applications was key, though development was dispersed over the Android, iOS and Windows Mobile platforms. A decision was made to focus on a web-based HTML5 application. Using the offline functionality of HTML5 and CSS3 a mobile application was quickly prototyped.

Deployment: Uganda

The Africa Urban and Water (AFTUW) sector of the World Bank approached the Taarifa project about a pilot with the Ugandan Ministry of Local Government (MoLG). The ministry wished to monitor local government projects based around improving community cohesion, public services and enterprise. A pilot in four districts was decided upon as part of the "Improving Systems for the Urban Poor" of AFTUW supporting two ministry led programs Community Driven Development (CDD) and Local Government Management and Service Delivery (LGMSD).

CDD is a match funding program where community members form groups around themes of entrepreneurship, farming and education. Funding is given in ratios of 2-5:1 and are aimed at directly improving the development of communities. LGMSD is a government program aimed at building capacity within government. For instance the building of council facilities and schools.

Figure 5: Reporting.

Figure 5 shows the reporting of a local government building in construction by civil servants. Traditionally the system of CDD and LGMSD was paper based. These forms were then posted from the areas to central government in Kampala. This drawbacks, first was the postal service of Uganda with idiosyncratic delivery. Second was the load placed upon the reporters. The complex nature of the questions posed by CDD and LGMSD posed difficulty to civil servants, who may not have the appropriate equipment to submit forms.

Civil servants reporting to these programs were selected for training, with AFTUW supplying Android based Hauwei ‘Gaga’ mobile phones - under $100 - as the hardware platform. Initially the custom forms of Taarifa worked well, with the participants able to submit information. However, when venturing into more remote districts the functionality of Taarifa inhibited reporting, specifically offline forms. While reporting offline is possible, it isn’t possible to change forms without connectivity. This requirement since entered the Github repository; the platform used for the project management of the Taarifa platform.

Improvements were identified by the pilot and were fed back into the Taarifa community. However the pilot in the four districts was deemed successful by MoLG and AFTUW, consequently the platform was rolled out to the 111 districts of Uganda. Currently the system is directly administered by MoLG, however they are activity seeking a devolution of control to the local districts. This in itself will be a large undertaking and one which potentially requires more structure than the Taarifa project in its current form can provide.

Evaluation and comparison of community software contribution

Given the way Taarifa has been developed, a question emerges as to how to best evaluate the community effort behind this development. To address this question, we need to inspect the contributions made to the open source project and try and make sense of these. It is important to remember that Taarifa was born as a fork of the Ushahidi project, and originally called Taarifa Web. This is the version in use in most deployments. A newer version, called Djangorifa, is a complete rewrite from scratch of Taarifa, using the Django framework in order to achieve a lightweight distribution. The rewrite also aims at freeing Taarifa from functionalities and features which were of use only to the goals for which Ushahidi was developed. In this section we will present a comparison of these three projects based on code contribution statistics.

Data collection methodology

All of the three projects, Ushahidi, Taarifa Web, and Djangorifa, are community open source project. They all use the same set of online tools and services in order to allow collaborationg: GitHub, an online code repository based on the git protocol. The protocol allows us to collect data which can be used to extract statistics. Using the `git log` and `git shortlog` commands, we were able to identify the number of commits, files changed, lines insert and lines deleted for each contributor. From this data we extracted information like the number of contributors, the lines committed, the number of commits, and calculated average contribution, standard deviation, and the relative quintiles. This methodology has its limitations, intrinsic to the type of projects we are discussing. For example, it must be remembered that the Taarifa Web project comes from a fork of Ushahidi, hereby sharing much of the codebase; in this analysis, we focus on the code contributed after the fork because we can then see how much code has been genuinely added. Also, we use quintiles analysis because we are talking about different amounts of code contributed: given that Taarifa embeds much of the Ushahidi codebase, having a quintiles analysis on the differential contribution allows us to compare the community involvement and interactions. The Djangorifa project was born as a single-person effort and is a relatively smaller project. Still, we add it for completenes and comparison. Although we cannot claim this is a complete analysis, it is helpful to identify trends and customs in community contributions. Data are summarised in Table 1.

Table 1: Summary of project statistics.

Param	**Ushahidi**	**Taarifa Web**	**Djangorifa**
contributors	71	12 (83)	5
commits	4017	667 (4684)	55
branches	8	6	4
lines of code	>220,000	>40,000 (260,000)	>35,000
average commits	56.58	7.53	11
std. dev. commits	154.66	5.06	15.35
average LOC inserted	13401.64	2935.92	5927.4
std. dev. LOC inserted	37496.32	9966.27	10553.19
issues (open)	240	21	11
issues (closed)	779	29	4

Ushahidi

A total of 71 developers have contributed to this project, which is a medium-large software project, also highlighted by the high number of lines of code (in excess of 240,000) and commits (over 4,000). With such a great number of contributors, we can see very large variations in their contributions. Despite the largest single contributor having committed almost 800 times with over 40,000 lines of code (LOC), the average is about 57 commits per contributor, with a very high standard deviation of 154.66; the same applies to the LOC, where we see an average of 13401.64 and a standard deviation of 37496.32. This shows that the contributions have been varied across the community, although we have a strong core of developers who have contributed a huge amount of LOC. An interesting analysis can be that considering the quintiles. For example, where commits are concerned, figure 6 shows how many contributors fall in each quintile. The quintile range is defined in the caption of the figure. It is evident that there is a relatively high amount of people only contributing a small number of times to the projects (20th percentile). The number then drops for the 40th percentile and slowly increases up to the 100th percentile. This is consistent with a community where there is a core of steady contributors collaborating with a galaxy of less involved contributors. However, if we apply the same analysis to the number of LOC contributed (see 7), we find that there is a certain balance in the number of contributors within each quintile. This suggests that the amount of effort each contributor is able to provide the community is similar, although sometimes spread over multiple commits. There is, however, a medium correlation (correlation coefficient of 0.49), between the commits and the LOC samples, which is consistent with an ongoing effort.

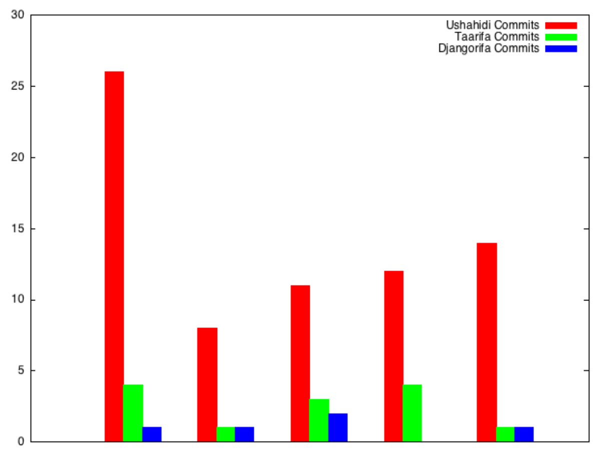


Figure 6: Quintile cardinality for commits; Ushahidi [1,2,5,34,777], Taarifa Web [3,5.8,8.2,12,18], Djangorifa [1.8,5,7,13.2,38]

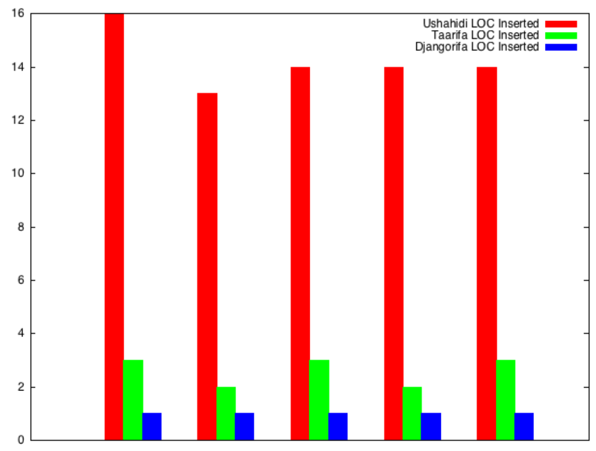


Figure 7: Quintile cardinality for number of lines of code inserted; Ushahidi [6,26.4,226.6,9706.4,229389], Taarifa Web [4,28,190.6,481.2,36098], Djangorifa [50.2,132.8,2087.6,8839.8,24427]

Taarifa Web

The Taarifa community is much smaller than the Ushahidi community, and the total amount of LOC is bigger only because Taarifa Web was originally forked out from Ushahidi. In fact, Taarifa adds about 40,000 LOC on top of the Ushahidi code base, coming to a total of 260,000. The number of contributors who worked exclusively to Taarifa is 12, coming to a total of 667 commits. Clearly, this implies that in its shorter life each contributor had fewer opportunities to commit code: the average per-contribute is 7.53 with a standard deviation of 5.06. This suggests more uniformity in the way Taarifa Web was developed and is in fact consistent with its inception at a hackathon: there is a relatively high number of people doing commits, rather than the massive differences we saw for Ushahidi. The story is a bit different when we analyse the LOC contribution. With an average of 2935.92 and a standard deviation of 9966, we can still see wild differences in the levels of contribution. This finding can be explained in a very simple way: Taarifa is a community effort were there is a core groups of developers, and most importantly a lead developer. The lead developer is most definitely an outlier, having contributed a whopping 36098 LOC. In this case, there is a clear concentration of many LOC in just a few commits, typical of hackathon developments. This is confirmed by the low correlation (0.28) existing between the commits and LOC samples. Visual confirmations of this can be found in figure 6 and 7.

Djangorifa

As stated before, we only add this for completeness. Djangorifa is a novel implementation of Taarifa based on a new framework and it is pretty much an ongoing effort. The data we see here are consistent with an ongoing personal developmental effort, especially an extremely low correlation coefficient (0.05) between the commits sample and the LOC.

Discussion

The most important lesson we gather by analysing these data is that we can distinguish an ongoing effort, spread over a large community, that of Ushahidi, from an development like that of Taarifa, which had an initial large contribution at a hackathon, upon which a small community then developed. Also, we are able to identify where there is a core of developers, a lead developer, or a single developer, as in the case of Djangorifa. Further analysis might lead us to identify trends in Taarifa, seeing when and where contributions are more likely to give a lasting effect on the community.

Summary and Future

This paper has discussed how the Taarifa project was started and how it was used in Uganda. Issues identified with the deployments include problems with civil infrastructure and communications in the country. While it is realistic to adapt the Taarifa platform to be resilient with regard to poor connectivity it will presumably be an issue which will need to be addressed in future iterations.

How can the Taarifa platform deal with an environment with no connectivity? We are currently assessing and piloting the use of Taarifa in Tanzania and in the United Kingdom, the adaption in these environments, with differing infrastructure and political will, remain outstanding. Taarifa as a group is currently looking towards formalising as an organisation. As an open source movement it can go so far, however as a loose collection of interested humanitarians the project can only go so far. For example documentation is an area which is in need of improvement, not just in requirements but user guides and manuals of use. Unfortunately "the code is the documentation" isn't an approach that the Taarifa project wishes to take.

As an organisation, formal structures and roles can aid in shaping the project. Requirements gathered in collaboration with users of the platform at the ministerial and local level could be investigated with the funding to explore those opportunities. Taarifa is an open source platform and project and is free to download and use. However the time and equipment spent on the project is costly. The community, however, has proved successful in providing enough motivation to the members to keep working on the project.

References

[1] E Hagen. Putting Nairobi's Slums on the Map. Development Outreach, 12(1):41–43, 2009.

[2] M Iliffe. When Government 2.0 Doesn't Exist : Mapping Services In The Developing World. In Proceedings of the 2011 AGI GeoCommunications Conference, 2011.

[3] Matthew Zook, Mark Graham, Taylor Shelton, and Sean Gorman. Volunteered geographic information and crowdsourcing disaster relief: a case study of the Haitian earthquake. World Medical & Health Policy, 2(2):7–33, 2012.

[4] Nathan Schurr, J Marecki, and M Tambe. The future of disaster response: Humans working with multiagent teams using DEFACTO. In Proceedings of the AIII Spring Symposium, 2005.

[5] K Lorincz, D J Malan, T R F Fulford-Jones, A Nawoj, A Clavel, V Shnayder, G Mainland, M Welsh, and S Moulton. Sensor networks for emergency response: Challenges and opportunities. Pervasive Computing, IEEE, 3(4):16–23, 2005.

[6] A Heijmans and L Victoria. Citizenry-Based & Development-Oriented Disaster Response. Centre for Disaster Preparedness and Citizens' Disaster Response Centre, 2001.

[7] L Gunawan, A. H. J. Oomes, M Neerincx, WP Brinkman, and H Alers. Collaborative situational mapping during emergency response. Interacting with Computers, 2009.

[8] P Meier. Do "Liberation Technologies" Change The Balance Of Power Between Repressive States And Civil Society? PhD thesis, Tufts University, 2011.

[9] Erica Hagen. Mapping Change: Community Information Empowerment in Kibera (Innovations Case Narrative: Map Kibera). Innovations: Technology, Governance, Globalization, 6(1):69–94, January 2011.

[10] Rebecca Goolsby. Social media as crisis platform: The future of community mapscrisis maps. ACM Transactions on Intelligent Systems and Technology (TIST), 1(1):7, October 2010.

[11] E Christophe, J Inglada, and J Maudlin. Crowd-sourcing satellite image analysis. In Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International, pages 1430–1433. IEEE, 2010.

[12] Leska S Fore, Kit Paulsen, and Kate O'Laughlin. Assessing the performance of volunteers in monitoring streams. Freshwater Biology, 46(1):109–123, 2001.

[13] B Al-Ani, G Mark, and B Semaan. Blogging in a region of conSSict: Supporting transition to recovery. In Proceedings of the 28th international conference on Human factors in computing systems, pages 1069–1078. ACM, 2010.

[14] T. N. Smyth, J Etherton, and M. L. Best. Moses: Exploring new ground in media and post-conflict reconciliation. In: Proceedings CHI 2010, 2010.

[15] Robert Cecil Martin. Agile software development: principles, patterns, and practices. Prentice Hall PTR, 2003.

[16] K. Mei Chen and C. Reed. OGC: Open GeoSMS Standard-Core. Technical report, Open Geospatial Consortium, 2012.