# StakeSource: Harnessing the Power of Crowdsourcing and Social Networks in Stakeholder Analysis

Soo Ling Lim*
Dept. of Computer Science
University College London
United Kingdom
s.lim@cs.ucl.ac.uk

Daniele Quercia
MIT SENSEable City Lab
Cambridge
USA
quercia@mit.edu

Anthony Finkelstein
Dept. of Computer Science
University College London
United Kingdom
a.finkelstein@cs.ucl.ac.uk

## ABSTRACT

Projects often fail because they overlook stakeholders. Unfortunately, existing stakeholder analysis tools only capture stakeholders' information, relying on experts to manually identify them. StakeSource is a web-based tool that automates stakeholder analysis. It "crowdsources" the stakeholders themselves for recommendations about other stakeholders and aggregates their answers using social network analysis.

## Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Requirements/Specifications

## General Terms

social network analysis, algorithms

## Keywords

stakeholder analysis, social networks, recommender systems

## 1. INTRODUCTION

Stakeholder analysis is a critical step in software engineering. Omitting stakeholders is one of the most common mistakes in software development [2] and the main cause for project failure [6]. As stakeholders are the source of requirements [3], an incomplete list of stakeholders gives rise to missing requirements, which in turn leads to the wrong product being built and a failed project. As such, proper stakeholder analysis – the process of identifying and prioritising stakeholders based on their influence in a project – is crucial to project success.

In our research reported in this ICSE [4], we have developed StakeNet, an approach for stakeholder analysis. In StakeNet, the experts *identify* stakeholders by asking them to recommend other stakeholders, *build* a social network of stakeholders from their recommendations, and *prioritise* the stakeholders using various social network measures. We have applied StakeNet to a large-scale software project – the Replacement Access, Library and ID Card project (RALIC) in University College London (UCL). Results show that StakeNet identifies a complete list of stakeholders, and prioritises the stakeholders accurately according to their influence in the project. Nevertheless, StakeNet requires experts to approach stakeholders individually to ask for recommendations. As such, it is costly for large projects with many stakeholders. An improvement would be a tool to support the process.

Existing tools provide little support in the actual *identification* and *prioritisation* of stakeholders; they merely *hold* and *process* the data provided by the experts in charge of stakeholder analysis. For example:

- **Stakeholder Analysis Matrix**[1]. The experts make a list of stakeholders and plot them against two variables on a matrix, such as power and interest, or importance and influence.
- **Stakeholder Circle**[2]. The experts enter stakeholder information such as name, role, significance, comments about the stakeholder, and whether the stakeholder is active. Based on the information, the tool generates graphs and reports.
- **Stakeholder Checklists** (e.g., the Onion Model [1] and the Volere Stakeholder Analysis Template[3]). The checklists contain a set of generic stakeholder roles (e.g., user and regulator). By referring to these roles, the experts derive project specific stakeholder roles (e.g., students and data protection officer).

Perhaps, in smaller projects, these tools may suffice. Nevertheless, in large-scale projects where no individual has the global perspective, the experts using these tools are bound to omit stakeholders. For example, RALIC used the checklist tool to identify stakeholders. Lead requirements engineer, Mary[4], was unaware that external library users would be affected, hence failed to identify them despite using the tool. As a result, UCL libraries remained with the old access

*Also with the University of New South Wales and NICTA.

[1]http://www.mindtools.com/pages/article/newPPM_07.htm
[2]http://www.stakeholder-management.com/
[3]http://www.volere.co.uk/templates.htm
[4]The names in this paper have been changed for reasons of privacy.

Citation: S. L. Lim, D. Quercia, and A. Finkelstein. StakeSource: Harnessing the power of crowdsourcing and social networks in stakeholder analysis. In Proceedings of the IEEE 32nd International Conference on Software Engineering, 2010 (in press).

control system. Mary is not to blame. In such a large project with more than 60 stakeholder groups, the odds were low for her to know about the external library users, unless Janet, the library systems manager, told her about Matthew, the memberships officer, who then revealed that external library users would be affected. An important question then arises: how do we make that happen?

The answer comes from a new approach that enables us to exploit the knowledge contained in diverse communities of people [7]. Crowdsourcing – or in this case, stakeholder-sourcing. Instead of having the experts approach stakeholders for recommendations, the tool should automatically source the stakeholders for help. Ideally, by using the tool, all the experts need to do is start off with an initial incomplete set of stakeholders, and the tool should automatically return a complete and prioritised list of stakeholders. This is what we set out to achieve with StakeSource described in the next section.

## 2. STAKESOURCE

StakeSource relieves the experts from the burden of stakeholder analysis by providing four features.

### Feature 1: Identify Stakeholders

> *StakeSource returns a complete set of stakeholders based on initial stakeholders that the experts provide.*

**Example**. StakeSource asks Mary to list down the stakeholders she knows about in the format <name, role, email address>. For example, Mary makes an entry <Janet, library systems manager, j.lib@ucl.ac.uk>. Based on the list of stakeholders that Mary enters, StakeSource returns a list of stakeholders and their roles, including external library users – the stakeholder Mary overlooked.

**How StakeSource does it**. StakeSource identifies stakeholders by asking stakeholders to recommend other stakeholders. To use the tool, the experts start by creating the project and entering the project details such as project description and scope items. StakeSource uses these details to inform stakeholders about the project when it asks for their recommendations.

StakeSource prompts the experts for initial stakeholder roles by asking them to list down the user, developer, legislator, and decision-maker roles in the project, using a wizard. For each role, the experts tell StakeSource about the initial stakeholders by entering their names, roles, and email addresses. StakeSource sends an email to each initial stakeholder, asking him or her to recommend other stakeholders. The email contains a link that will bring the stakeholder to a recommendation form (Figure 1) with hints about how to recommend. Each stakeholder recommendation is in the format <name, role, salience (the level of influence a stakeholder has on the project), email address>. If a stakeholder is aware of a role but is not aware of the stakeholders with that role, he is allowed to recommend only the role. In that case, StakeSource associates that recommendation to all the stakeholders with the same role who are identified so far.

Each time a new stakeholder is recommended, StakeSource sends an email to the new stakeholder requesting recommendations about other stakeholders. This procedure is called the snowballing technique [5], whereby the group of stakeholders identified by StakeSource builds up like a snowball

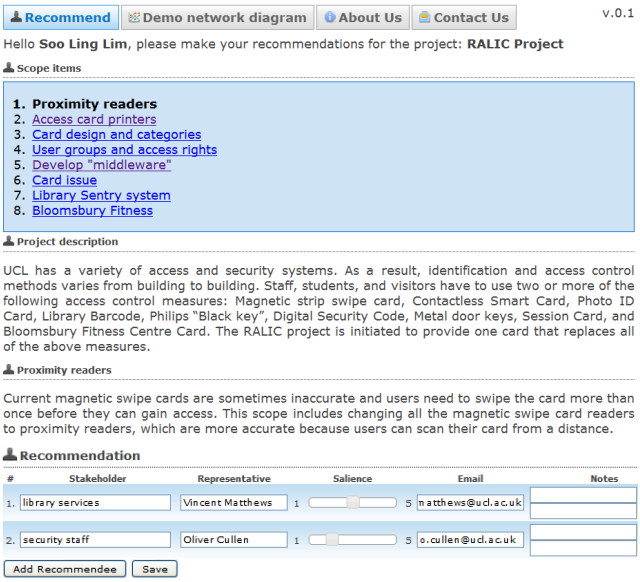


**Figure 1: Recommendation form.**

rolled down a hill. Eventually, few additional stakeholder roles are identified in each round of requests. When no additional roles are identified in one round of requests, the experts stop the process, and StakeSource returns a list of identified stakeholders and their roles.

### Feature 2: Prioritise Stakeholders

> *StakeSource prioritises stakeholder roles and stakeholders by their characteristics using social network measures [4] such as betweenness centrality, closeness centrality, and PageRank.*

**Example**. Mary wants to mediate between stakeholders with conflicting requirements. She chooses the *betweenness centrality* measure (Figure 2 Panel A) to prioritise brokers between disparate clusters of stakeholders. Among the brokers, StakeSource identifies "Estates" as a good broker, and within Estates, Martin is a better broker than Richard.

**How StakeSource does it**. StakeSource aggregates each stakeholder's private judgements about a stakeholder's influence in the project into a collective decision. Using the stakeholders' recommendations, StakeSource draws a social network with the stakeholders as nodes, and their recommendations as directed edges: *S1* links to *S2* if *S1* believes *S2* to be a stakeholder. StakeSource provides two levels of social network: *scope item* and *project*. At the *scope item* level, the salience in each recommendation determines the weight of the link. At the *project* level, StakeSource combines scope-level recommendations such that if *S1* recommends *S2* in $N$ number of scope items, $N$ determines the weight of the link.

StakeSource prioritises the different stakeholder characteristics for a stakeholder $S$ using different *social network measures* [5] as follows.

- *Betweenness centrality* ranks $S$ by summing the number of shortest paths between other pairs of stakeholders that pass through $S$.

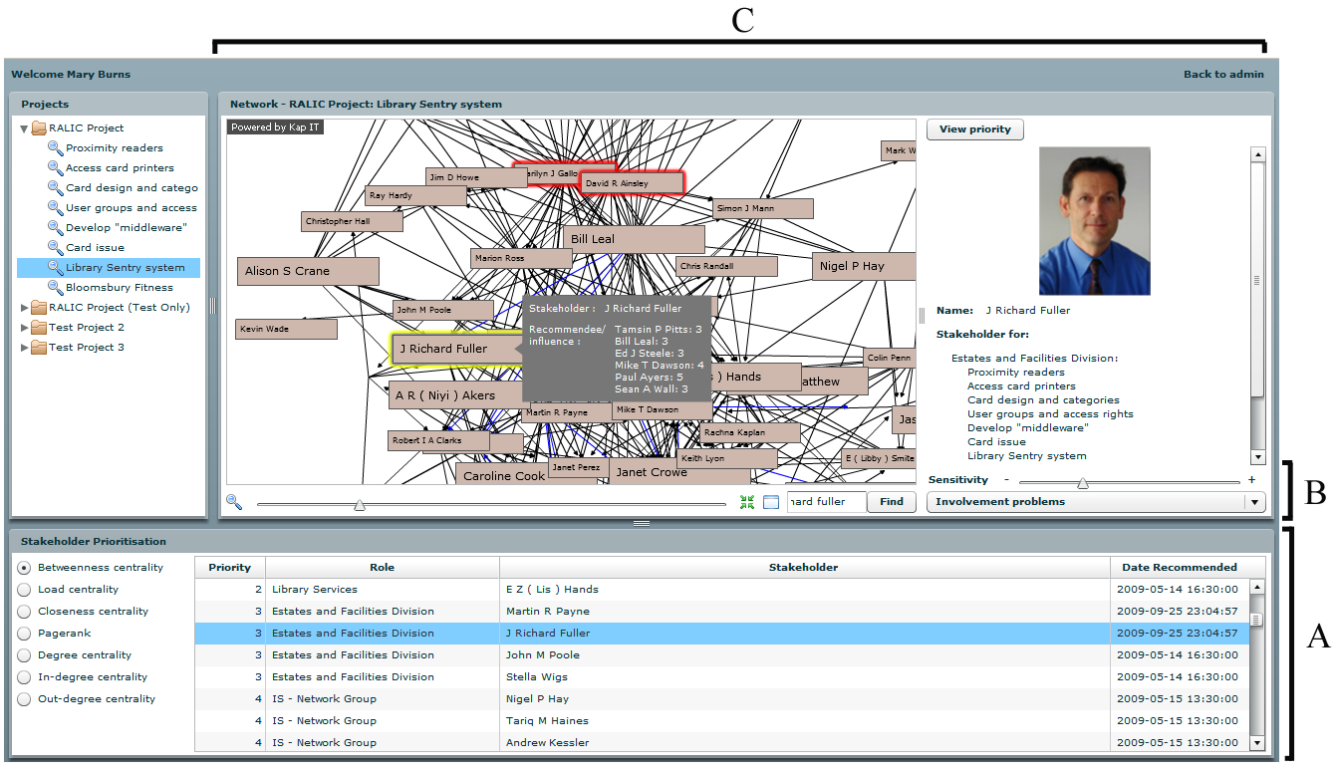


**Figure 2: The three panels (A, B, and C) of StakeSource's user interface.**

- *Load centrality* ranks $S$ by the total amount of recommendations passing through $S$ from adjacent stakeholders.
- *Closeness centrality* ranks $S$ based on $S$'s distance to all other reachable stakeholders from $S$.
- *PageRank* ranks $S$ in terms of $S$'s relative importance to all other stakeholders. The rank is defined recursively and depends on the number and rank of all the stakeholders that recommend $S$.
- *Degree centrality* ranks $S$ based on the number of direct connections $S$ has, both to and from other stakeholders.
- *Out-degree centrality* ranks $S$ based on the number of recommendations that $S$ makes and the recommendations' weights.
- *In-degree centrality* ranks $S$ based on the number of recommendations that $S$ receives and the recommendations' weights.

For each measure, StakeSource produces a prioritised list of stakeholder roles, and for each role a prioritised list of stakeholders. To improve the quality of prioritisation, the experts can merge similar recommended roles (e.g., PhD student and research student), and different names referring to the same person.

## Feature 3: Identify Potential Problems

*StakeSource highlights stakeholders who may have problems with their involvement or communication.*

**Example**. StakeSource tells Mary that security head Nicholas may have involvement problems (e.g., lack of time or interest in the project) and more effort is required to keep him engaged. Mary adjusts the sensitivity slider in Figure 2 (Panel B): the more sensitive the tuning, the more stakeholders with potential problems are returned. This helps Mary explore stakeholders with potential problems in the various states of sensitivity.

**How StakeSource does it**. For potential involvement problems, StakeSource finds stakeholders who rank high in *degree centrality* but low in *betweenness centrality*. For potential communication problems, StakeSource finds stakeholders who rank high in *betweenness centrality* but low in *closeness centrality*. The slider determines the allowable differences between the rank. When the slider is in a more sensitive mode, the allowable difference is less.

## Feature 4: Display Stakeholder Information

*For each stakeholder, StakeSource displays the following information: name, role, photo, the scope items they are recommended for, and comments from other stakeholders. It also visualises the stakeholder's position on the network, who they recommend, and their rank as different kinds of stakeholders.*

**Example**. Using the search function, Mary finds Richard's node and the network diagram in Figure 2 (Panel C) highlights his node. Mary clicks on the node to pop up Richard's profile and find out what he looks like, his role in the different scope items, unavailability on Wednesdays, and rank as a different kind of stakeholder. Hovering over Richards's

node with the mouse pointer reveals his recommendations. The prioritisation panel shows that Richard is a more suitable stakeholder than John for the "Estates" role. From the network visualisation, Mary is able to see that Richard sits between two clusters, suggesting that he is a good broker.

**How StakeSource does it**. When a stakeholder recommends someone, he can also add public and private notes about the person. The network diagram displays each stakeholder as a node, with his outgoing and incoming recommendations, and a rank number for each stakeholder characteristic. Clicking on the node highlights the stakeholder's row in the prioritisation panel and displays the stakeholder's details.

# 3. IMPLEMENTATION

For StakeSource's implementation, we have the following key requirements, which are informed by our recent research [4].

- **Widely available and easy to access**. StakeSource works when a sufficient number of stakeholders contribute with their recommendations.
- **Simple and intuitive to use**. The ease of recommendation is vital to encourage stakeholders' contribution.
- **Easy to make recommendations without compromising quality**. Closed-ended recommendations where stakeholders are provided with an existing list of stakeholder roles are easier to complete but return a less complete set of roles.
- **Interactive stakeholder analysis UI**. The experts should be able to interact with the UI to learn about the stakeholders (e.g. by clicking on a stakeholder's node in the network, the prioritised list should highlight the stakeholder's row).

Based on the key requirements, we have made the following design decisions for StakeSource.

- **Web-based**. To make the tool widely accessible, we have implemented it as a web application using standard web technologies such as HTML, CSS, XHTML, PHP, and JavaScript. We have selected MySQL for data storage.
- **Standard interface with simple terms**. We have implemented the recommendation forms (Figure 1) using standard survey interface from the Smarty Template Engine[5]. On top of that, we have used widely known terms whenever possible. For example, we have chosen the term "level of influence" over "salience" because beta testing revealed that "level of influence" is easier for stakeholders to understand. Finally, we have supplied tool tips and pop-up help for difficult terminology. Future work involves translating the technical terms for the measures in the stakeholder prioritisation feature to correspond to specific end-user goals.
- **Autocomplete**. For the stakeholder role entry, we have opted for autocomplete using jQuery rather than a drop-down list.
- **Existing components**. We have implemented the stakeholder analysis user interface (Figure 2) in Flex, and reused two well-established software components.
  (a) **Kap Visualizer**[6]. We use the Flex Visualizer for its customisable and interactive network visualisation.
  (b) **NetworkX**[7]. We use the Python package for its implementation of the social network measures.

StakeSource is available from our project website[8].

# 4. CONCLUSIONS

The lead requirements engineer, Mary, omitted a major stakeholder, causing the libraries in University College London to remain with the old access control system. Had Mary used StakeSource for RALIC, the project would have achieved its objective of a unified access control system across campus.

StakeSource is a simple but surprisingly powerful and useful tool. It proposes a shift from current practices where a team of experts conduct stakeholder analysis, to a crowdsourcing approach that involves all stakeholders. In doing so, it reduces the experts' workload and lowers the likelihood of overlooking stakeholders. The tool has now been used in the UCL Admissions System Project[9], a large-scale software project involving more than 70 stakeholders. We plan to use the tool for projects in other fields, such as environment policy planning.

# 5. ACKNOWLEDGMENTS

We thank Jason Lau for his contribution in the software development, Abdigani Diriye and Enzian Baur for their advice in the user interface, and Peter Bentley for his feedback on the paper.

# 6. REFERENCES

[1] I. Alexander and S. Robertson. Understanding project sociology by modeling stakeholders. *IEEE Software*, 21(1):23–27, 2004.
[2] D. C. Gause and G. M. Weinberg. *Exploring Requirements: Quality Before Design*. Dorset House Publishing, 1989.
[3] M. Glinz and R. J. Wieringa. Stakeholders in requirements engineering. *IEEE Software*, 24(2):18–20, 2007.
[4] S. L. Lim, D. Quercia, and A. Finkelstein. StakeNet: Using social networks to analyse the stakeholders of large-scale software projects. In *Procs. of the 32nd Int. Conf. on Soft. Eng. (to appear)*, 2010.
[5] J. Scott. *Social Network Analysis: A Handbook*. Sage, 2000.
[6] Standish Group. *The CHAOS Report*, 1994.
[7] J. Surowiecki. *The Wisdom of Crowds*. Anchor, 2005.

[5]http://www.smarty.net/

[6]http://lab.kapit.fr/display/kaplabhome/Home
[7]http://networkx.lanl.gov
[8]http://www.cs.ucl.ac.uk/research/StakeSource/
[9]http://www.ucl.ac.uk/isd/community/projects/azlist-projects