something of a dark art, but beyond the overall structure of a site – which is imposed by default by the above platforms – there are general tips on how to phrase your written content and what styles of heading and caption work best (on the whole: be concise, plain, but descriptive). All of which makes a lot of sense. If you set-up a website, chances are you'd like as many people as possible to find it through searches for relevant related content.

But step back a moment. Is it always a good thing to be writing content and structuring a site primarily so it can be easily parsed by a program? What about all that freedom of expression and artful presentation? In the week I was playing around with WordPress, *New Scientist* coincidentally featured a piece called "Welcome to the 'algoworld'", which paints a picture of a world where misunderstood entities known as "algorithms" are running amok. But despite the alarmist vision, there were several sentences that chimed with my thoughts, such as this observation on recommender systems (such as Amazon uses): "The danger is that such an algorithm can create a monoculture. But this is not the way culture works – it is actually much spikier, much less predictable". And this, reiterating the above idea that search engines are the new patrons of our creativity: "It used to be that you wrote news for how people read – now it's written for how machines read ... It is shaping our expression and behaviour."

Requirements engineering is a discipline filled with frameworks and templates, each intended to elicit or capture human assumptions, intentions, and whims in efficient or exhaustive ways. There's no denying, generally speaking, that such frameworks are a boon to the requirements analyst or engineer. But are they always to the benefit of the activity? Do any of our readers have stories to support a view one way or the other?

I'd like to invite you share thoughts and observations: How important is it for a particular requirements framework or template – or notation, or modelling trope – to allow flexibility? Or, on the other hand, is it always more beneficial for tools of the requirements trade to impose artificial constraints or structure on the domain of human expression, though it may be rife with vacillation? Obviously, there's a balance but where does that balance lie?

If you have an opinion, observation, or anecdote please contact the editor – william.heaven@gmail.com – or leave a comment for this edition of *RQ* on the RESG website!

William Heaven

RE-flections

The Forgotten Keystone of Requirements Engineering

Soo Ling Lim

A newly launched expense system was rejected by its users. They were unaware of the project and the system was not meeting their needs. The system had to be redeveloped.

A railway timetabling website project was delayed for half a year. The website was completed on time, but a week before its scheduled release, the project manager found out to his horror that the new timetables needed approval. The approval took six months and should have been sought at the start of the project.

A team of developers spent the first year after their software was released handling change requests from users who had been ignored during requirements elicitation.

A medical software was banned from sales because it had not been approved by the health authority.

All these projects suffered from the same problem – their stakeholders were not identified correctly at the start. By stakeholders, I mean individuals or groups who can influence or be influenced by the outcome of

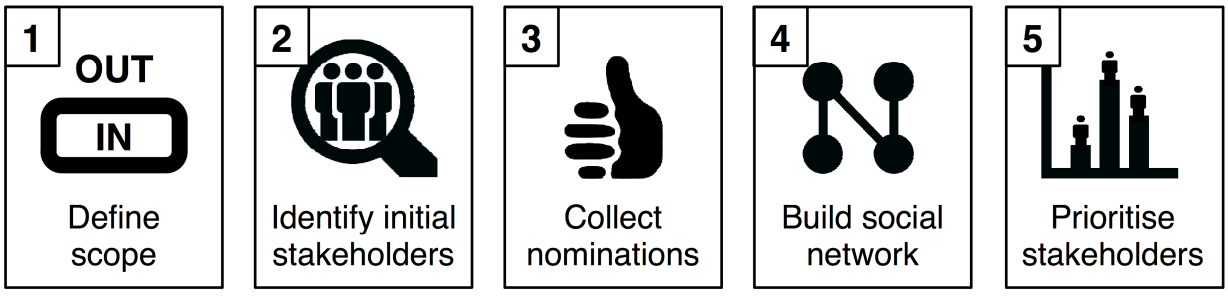

Fig. 1: Stakeholder Identification in Five Steps.

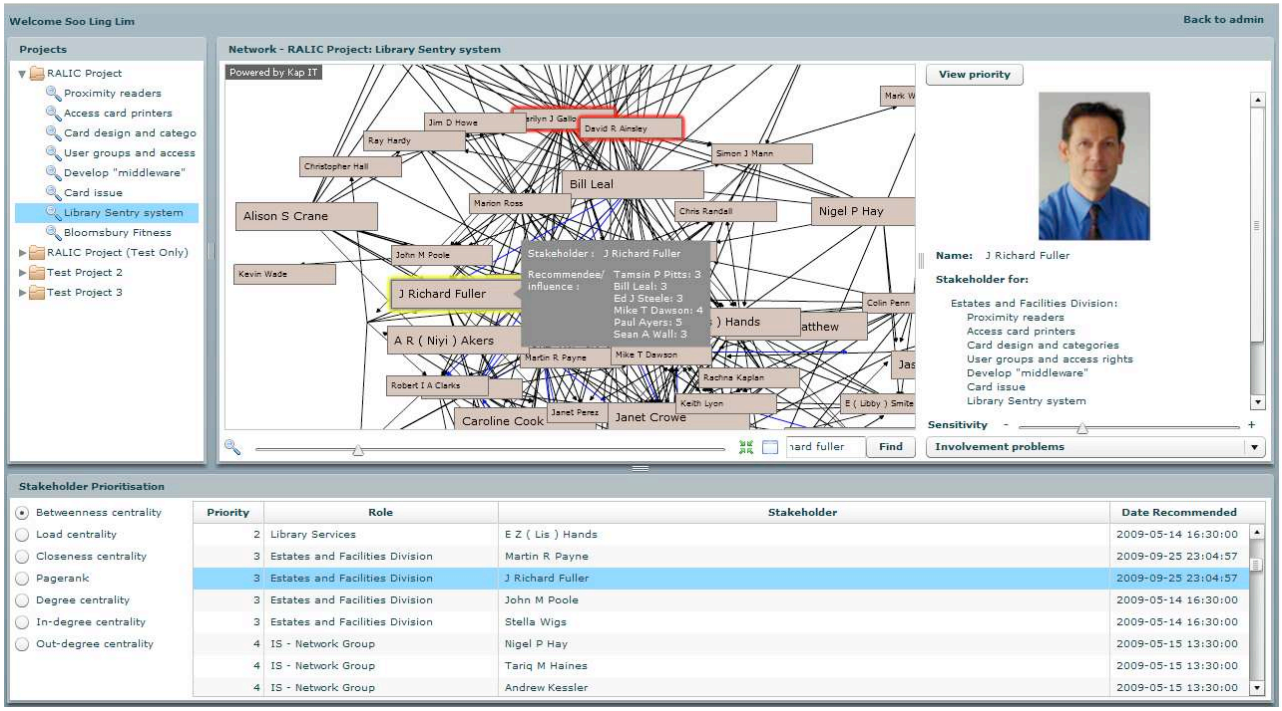

Fig. 2: Stakesource Network View

the software project.

Stakeholders play a central role in requirements elicitation. Requirements and project constraints come from stakeholders, so omitting stakeholders gives rise to missing requirements, which leads to building the wrong product. Some projects have only a handful of stakeholders; others can have hundreds or even hundreds of thousands of stakeholders. Some stakeholders have more influence, knowledge, or interest in the project than others. As such, the identification of suitable stakeholder representatives is crucial.

It is a mystery to me why stakeholders seem to be ignored or forgotten in RE. In a recent RESG poll by Camilo Fitzgerald [1], no one was interested in stakeholder identification! Yet in most of the projects I have seen, stakeholder identification is at worst non-existent and at best reactive – the analyst starts with an incomplete list of stakeholders and tackles new stakeholders as they appear throughout the project. In most cases, this reactive approach slows down the process and adds rework. But when an important stakeholder appears too late in the process, we have yet another horror story like the ones above.

This problem is well-known and has been described by many of the leaders of the field. Anthony Finkelstein listed stakeholder identification as one of the unsolved problems in requirements engineering in his 2004 AGM talk [2]. Ian Alexander and Suzanne Robertson held workshops called Stakeholders Without Tears [3]. I believe it is time for the RE community to tackle the problem of stakeholder identification head on – and that is why my research has focussed on this topic.

Social Networks For Stakeholder Analysis

My research uses ideas from the field of social network analysis. Stakeholders are socially related to one another. While each stakeholder only has a partial view of the project, together the group of stakeholders may hold a more complete picture than an individual expert. This new method involves all stakeholders in the stakeholder identification process. The analyst asks stakeholders to nominate other stakeholders, builds a social network of stakeholders, and prioritises the stakeholders using social network algorithms [4].

The method has five steps and is illustrated in Fig.1.

1. Define the scope of your project. Scope describes the boundary of the project, so that you know which stakeholders should be involved.
2. Based on this scope, identify an initial list of stakeholders.
3. Ask each stakeholder to nominate other stakeholders. Then, ask each newly identified stakeholder to nominate other stakeholders. This technique is also known as the snowballing technique. The identification of new stakeholders may result in the need to modify the scope. Feel free to do so and remember to adjust the stakeholder list accordingly.
4. Build a social network of stakeholders using the information collected in the previous step. Each node in the network represents a stakeholder.

Each link represents a nomination such that Node A links to Node B if stakeholder A nominates stakeholder B.

5. Apply social network algorithms to the stakeholder network. These algorithms rank stakeholders based on their position on the network (similar to how Google PageRank ranks websites). This step produces a prioritised list of stakeholders for your project.

The method was first tested on a project with a complex stakeholder base. The results were very exciting – I found that the method identified a comprehensive list of stakeholders. Not only that, but prioritisation using social network algorithms was more accurate compared to prioritisation from individual experts.

I was pleased at how well the method was received. Software companies were inviting me to talk about the method and share my experiences. Developers wanted their managers to use the method, and I was getting requests from project managers for a tool support. An IT director wanted to make the method a standard in his organisation. It was all very exciting. But manually collecting hundreds of nominations and processing them using the social network algorithms was something I could only do to test the method. For the method to become mainstream, most of the steps in the method can and should be automated.

My solution was StakeSource [5].

Stakesource

StakeSource is a Web2.0 tool that uses social networks and crowdsourcing for stakeholder identification [6]. It "crowdsources" the stakeholders for nominations, using their stake in the project to motivate their response. By using StakeSource, you only need to provide project details and initial stakeholders. StakeSource contacts the initial and newly identified stakeholders for nominations, builds the stakeholder network, applies the social network algorithms, and returns a prioritised list of stakeholders. You can access up-to-date information about the stakeholder network and prioritised list of stakeholders at any point during the process (see Fig. 2).

There was a lot interest in StakeSource. Less than a year of the prototype going live, more than ten projects had successfully used StakeSource – not bad for a system developed at the end of a PhD!

StakeSource continues to generate interest. We are meeting executives from various telecommunications, business analytics, and software companies. Most recently I have been invited to present StakeSource in the Ready-Set-Transfer game show at RE'11 organised by Jane Cleland-Huang and Daniela Damian. The game show will follow the format of Dragon's Den, with a panel of industry practitioners as the dragons.

Stakeholders are forgotten no longer in RE. While my work may just be the first steps towards a solution in stakeholder identification, I hope it will stimulate fresh interest in this important area.

[1] Requirements Quarterly 55 (August 2010)

[2] Anthony Finkelstien, "Unsolved Problems in Requirements Engineering", keynote at RESG AGM 2004.

[3] Ian Alexander and Suzanne Robertson (2004). Understanding project sociology by modeling stakeholders. *IEEE Software* 21(1), pp. 23-27

[4] Soo Ling Lim, Daniele Quercia, Anthony Finkelstein (2010). StakeNet: using social networks to analyse the stakeholders of large-scale software projects. *ICSE* (1) 2010, pp. 295-304

[5] www.stakesource.co.uk

[6] Soo Ling Lim, Daniele Quercia, and Anthony Finkelstein, StakeSource: harnessing the power of crowd-sourcing and social networks in stakeholder analysis. *ICSE* (2) 2010, pp. 239-242.

Soo Ling Lim is a Research Associate in the Department of Computer Science, University College London. She has a PhD in computer science and engineering from the University of New South Wales in Australia, and a bachelor of software engineering with first class honours from the Australian National University. Before her PhD, she worked as an ERP analyst programmer and a SAP consultant at the Computer Sciences Corporation and as a software engineer at CIC Secure in Australia. She specialises in stakeholder analysis, requirements elicitation, prioritisation, and change management, and offers her services as a consultant in these areas.

Soo Ling can be contacted at s.lim@cs.ucl.ac.uk.

Upfront Analysis: The Payoff

Camilo Fitzgerald

The question of how much upfront RE analysis to perform is one of considerable complexity that has always plagued software projects – proposed answers have come from both ends of the spectrum. Research in requirements engineering typically suggests that more needs to be done [1], and that the cost ratio of fixing defects at the requirements level to later stages is between 5 and 10 for smaller projects and between 10 and 100 for larger ones [2]. In practice, however,