u-Help: Supporting Helpful Communities With Information Technology

(Extended Abstract)

Andrew Koster
Inst. of Informatics, UFRGS
Porto Alegre, Brazil
akoster@inf.ufrgs.br

Jordi Madrenas
IIIA-CSIC
Bellaterra, Barcelona, Spain
jmadrenas@iiia.csic.es

Nardine Osman
IIIA-CSIC
Bellaterra, Barcelona, Spain
nardine@iiia.csic.es

Marco Schorlemmer
IIIA-CSIC
Bellaterra, Barcelona, Spain
marco@iiia.csic.es

Jordi Sabater-Mir
IIIA-CSIC
Bellaterra, Barcelona, Spain
jsabater@iiia.csic.es

Carles Sierra
IIIA-CSIC
Bellaterra, Barcelona, Spain
sierra@iiia.csic.es

ABSTRACT

When people need help with day-to-day tasks they turn to family, friends or neighbours to help them out. Despite an increasingly networked world, technology falls short in supporting such daily tasks. **u-Help** provides a platform for building a community of helpful people and supports them in finding volunteers for day-to-day tasks. It relies on three techniques that allow a requester and volunteer to find one another easily, and build up a community around such provision of services. First, we use an ontology to distinguish between the various tasks that **u-Help** allows people to provide. Second, a computational trust model is used to aggregate feedback from community members and allows people to discover who are good or bad at performing the various tasks. Last, a flooding algorithm quickly disseminates requests for help through the community.

Categories and Subject Descriptors

Computing Methodologies [**Artificial Intelligence**]: Distributed Artificial Intelligence

Keywords

P2P application; agent-aided community building

1. INTRODUCTION

The web has been evolving into a social space enabling individuals to be pulled opportunistically into peer communities to achieve both personal and group goals. This has been possible due to the widespread adoption of software applications that support and facilitate basic group interaction such as file sharing, instant messaging, blogging, or social networking. These sorts of applications depend more on social conventions that arise within the community that uses them than on the particular features offered in the first place.

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013)*, Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.

But, despite the success of certain well-designed social software platforms such as Flickr, MySpace, LinkedIn, Facebook, or Twitter, current social applications do not allow a community to form around a specific need. Particularly, users who are looking for either new, partially formed, or already standing communities whose interactions are more domain specific, need software tools that support the evolving community.

In this paper we present **u-Help**, a software application that provides a fully distributed platform for building and maintaining a local community of people helping each other with their day-to-day tasks. The aim of **u-Help** is twofold. The first is straightforward: we aim to provide an application that finds a trustworthy member of the community who will help with the required task. The second aim is longer term: as the community evolves, it may have different rules and requirements. As such, maybe the tasks to be performed change, the way in which members' actions are evaluated change. The aim is to provide the tools to change the **u-Help** application together with the users. In this paper we focus exclusively on the first aim: we describe how the **u-Help** application supports a helpful community.

u-Help starts from the pre-existing social relationships between users. We thus assume the existence of a social network (identities and connections may be imported from various other social networking applications). To allow requests for help to propagate quickly through the network, we use a flooding algorithm. After a volunteer has been found and performs the task, the **u-Help** application allows the requester to evaluate that volunteer's performance. Such evaluations are aggregated in a computational trust model. Finally, we rely on semantic techniques to describe the tasks formally and define a similarity measure between them, in order to use evaluations of a volunteer performing one task to estimate his performance at another task.

2. THE U-HELP APPLICATION

The **u-Help** application provides a platform to build and support a community to which members can turn to find help with day-to-day tasks. Underlying this application is the existing social network data. The users of **u-Help** are represented as nodes in a graph and social relationships between the users are the edges of the graph.

When a member of the community needs help for performing a certain task, the *flooding algorithm* propagates this request, starting with that member's direct neighbours in the graph and flooding the request out from there until a satisfactory volunteer for the task is found. The flooding algorithm relies on the community member's trust evaluations of one another to decide whether to forward the request or not. Such trust evaluations are calculated automatically by the *computational trust model*. The trust model, in turn, relies on the *semantics of the task* to find similar tasks, when little to no feedback is available for a task.

2.1 Tasks

The tasks that **u-Help** finds volunteers for are domain-specific and depend on the community's needs. An example of such a community is a group of parents who coordinate tasks such as picking up their children from school (or other places) and babysitting them. The semantics of such tasks are given in a formal ontology, using a logical language. This allows us to define the similarity of the tasks using a variety of methods. For instance, for the example community above, we use an activity meronomy for the different types of activities users need help for, combined with a taxonomy for children, describing different age groups. These two representations require different similarity measures. We use Li et al.'s similarity measure [1] for the children, and the OpinioNet algorithm [2] for propagating trust evaluations over similar activities.

2.2 Trust Model for u-Help

Every member of the community maintains his own trust evaluations of other members of the community. These trust evaluations are task-dependent, with which we mean that a user's evaluation of another may change, depending on the task for which he is evaluated. This trust evaluation is crucial in the functioning of the flooding algorithm, which only forwards a request for help to a neighbour if the trust level in that neighbour is sufficiently high.

The flooding algorithm, which we describe in detail in the next section, relies upon each client having a trust evaluation of its peers for any possible task. Because peers may not have evaluations for all tasks, we take the similarity between tasks into account and allow trust to carry over, to a certain extent, to similar tasks. Additionally, some tasks may be more sensitive than others and require more trustworthy agents to perform them. Therefore we propose that users may specify a minimum trust level for each task.

The trust evaluation in a peer is calculated as a weighted mean over all past experiences and communicated evaluations; the weight is the similarity between two tasks.

2.3 Flooding Algorithm

The flooding algorithm is the core computational process in the **u-Help** platform. It ensures requests for help are disseminated through the community. When someone wants to request help with a specific task, the flooding algorithm sends this request to that person's neighbours in the social network graph (i.e. its friends) and from there it continues to flood through the network. This is similar to a number of other algorithms designed for rapidly disseminating a message through a graph, most prominently the Gnutella algorithm for P2P file sharing. The main difference between existing approaches and the flooding algorithm discussed here is that the decision to stop forwarding the request is made primarily based on trust, rather than on other things, like the number of hops the request has made through the network or the time passed since the initial request was made.

The reason for this is that we not only want to find someone willing to volunteer for a task, but he must also be trustworthy when performing this task. For calculating trust along a path, we assume that trust has some transitive properties, and thus we can use a T-norm to ensure trust is monotonically decreasing along a path. The flooding algorithm stops propagating the request when the cumulative trustworthiness of a node falls below a certain threshold τ.

Concerning efficiency, the flooding algorithm is not affected by loops in the graph as flooding is stopped when the node itself is found in the request path. In the worst case, the number of messages is $\sum_{p \in loop\_free\_paths(G)} length(p)$, which can be exponential in the number of nodes. Despite this worst case, simulations show that in realistic scenarios, the flooding algorithm performs well.

In addition, we note that, if needed, the algorithm could be changed to put a limit on the number of hops by taking a maximum number of hops into account, in addition to considering the trust level. This would make our algorithm more similar to the Gnutella flooding algorithm, but it would still take trust into account.

3. DISCUSSION

In this paper we have briefly introduced the **u-Help** application in its capacity as a platform for finding help within a community. However, as we said in the introduction, there is a second aim behind its development, to study how a community evolves and the methods we presented need to be augmented or changed over time. A specific extension that we intend to make in the **u-Help** application is the design of a charter in which the community's norms are described. The community needs some way of proposing such norms, and deciding which norms are adopted. Additionally, there must be some way of measuring norm compliance and identifying norms which need revising.

Acknowledgements

This work is supported by the Generalitat de Catalunya grant 2009-SGR-1434, the CBIT project (TIN2010-16306), the Agreement Technologies project (CONSOLIDER CSD2007-0022, INGENIO 2010), the ACE ERA-Net project and the PRAISE project.

4. ADDITIONAL AUTHORS

Additional authors: Angela Fabregues, Dave de Jonge and Josep Puyol-Gruart (IIIA-CSIC and Universidad Autònoma de Barcelona), and Pere Garcia-Calvés (IIIA-CSIC).

5. REFERENCES

[1] Y. Li, Z. A. Bandar, and D. McLean. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):871–881, 2003.

[2] N. Osman, C. Sierra, and J. Sabater-Mir. Propagation of opinions in structural graphs. In *Proc. of ECAI 2010*, volume 215 of *Frontiers in AI and Applications*, pages 595–600. IOS Press, 2010.