Socially-Augmented Software
Empowering Software Operation Through Social Contacts

Raian Ali, Nan Jiang, Huseyin Dogan
Bournemouth University, UK
{*rali,njiang,hdogan*}@*bournemouth.ac.uk*

***Abstract*—It is commonly agreed that a network of contacts can be an invaluable resource for an individual to obtain information and carry out certain tasks. However, this is still unexploited by software (i.e., contacts are not engaged when software acts to achieve the requirements of a user). Till recently, contacts and social relationships were hard to capture and manage by software and, thus, it was infeasible to use them to empower the operation of software. The emergence of social networks made it possible to realise a new kind of software-mediated social relationships which would enable what we call "socially-augmented operation". In this position paper, we propose socially-augmented software, which can benefit from network of contacts of users including individuals and organizations to fulfil their requirements. We articulate our view and discuss fundamental concepts of this new paradigm. A set of research challenges to better enable socially-augmented operations in software development will be also presented in the discussion.**

***Keywords*-Social Informatics, Software Engineering, Requirements Engineering, Social Contacts**

I. Introduction

In real life, people often need to meet certain requirements and achieve certain goals through collaborating with some of their contacts. Suppose you will be relocated to a new place and you need to find accommodation there. A friend of you who lives there could provide you some information to help you choose an area to move to. Even more, your friend can view a local property on behalf of you. In this scenario, the main characteristic of this collaboration is its voluntary nature. That is, contacts are not bounded to certain roles they actually have in an organizational environment where they have to fulfil requests as part of their responsibilities, e.g., customer service and IT desk. Instead, it is purely based on their will to interact and cooperate with each other to work together for overcoming the boundary of traditional business. Traditional software engineering is still limited in this hence it is necessary to call for novel approaches on eliciting, specifying and managing the power of contacts and integrating it with software.

The reliance on social contacts, in certain cases, is very practical and probably inevitable when it is relatively difficult to fulfil the entire requirements of an individual through solely automated means. The power of software could be augmented when it is integrated with the perception and rationale of humans and therefore social contacts. The basic idea of recent initiatives on social computer (e.g., [1]) is that machines are empowered by people brains and problem solving skills. However, there is still lack of engineering approach to get that implemented. This includes the consideration of the social relations between people which affect who can be asked to collaborate. Moreover, it is unclear what services to get from people and how they relate to the problem and users requirements. In fact, augmenting software with people is challenging so such consideration and restrictions need to be carefully designed to avoid discouraging or confusing users when interacting and cooperating with software. It should also note that allowing a freestyle interaction could lead to useless and even harmful solutions causing for example information overload.

The difficulty of capturing and managing social connections within the boundary of a computer-based system is one of the main reasons why this "empowerment of software operations through contacts" has not been made realistic until recently. With the increasing popularity of social networks, it provides some grounds to establish such integration through latest technologies. Moreover, a large number of people have already developed the mind-set necessary to manage their network of contacts using software-mediated social networks. This maximizes the potential of socially-augmented software.

Socially-augmented software which acts on behalf of users and communicate with their human contacts has to be dynamic so that dependency on contacts is socially-aware. The participation of contacts to empower software acting on behalf of a user is voluntary and, thus, its success is not always guaranteed. This is different from adaptive software, which is characterized by the ability to change behaviour when changes occur in its environment [2] as the goal of adaptation is to ensure that the requirements of users are fulfilled correctly and efficiently. Adaptation in socially-augmented software, however, is driven by the current status of both the ability of software to autonomously fulfil user requirements and the availability and capability of social contacts, which is not typically considered in the literature of adaptive system.

The engineering of socially-augmented software introduces new challenges and requires novel approaches. The main reason is that of the openness of such system and the inability to predict all possible variations of its social surrounding. Users are diverse and their social preferences are hard to predict and capture. So we need to develop platforms to allow users own interaction styles rather than dictating certain forms of interactions. The border between the role of a user and a software developer is thus blurred here: users need to express their social regime and software should provide them with tools to tailor it instead of restricting them to pre-designed policies and rules [16].

In this paper, we introduce the concept of socially-augmented software. Then, we focus on the requirements engineering activity of this new software category and study the integration of the power of contacts in boosting software to reach a user's requirements. Requirements are understood by people and hence, this level of abstraction is a natural place to start engineering social augmentation. We discuss a set of basic requirements artefacts which directly relate to the operation activities of socially-augmented software. They are: the activation of requirements, the quality assessment of the different solutions for requirements achievement and, the execution of a certain set of operations to enact a solution. Finally, a set of challenges will be considered and a research agenda for future work will be proposed.

II. Socially-Augmented Software

Traditionally, business software is designed to reach certain business requirements in the first place (e.g., book a flight and find the best place to stay in). Recently, social software focuses on supporting user interactions and relationships management by automated means. Socially-augmented software, which weaves between the power of contacts and the power of machines, is an integration of these paradigms. This is technically challenging as the domain of social contacts cannot be fully perceived and predicted by the analyst and it is, moreover, not under any control of an implemented system. For these reasons, socially-augmented software needs to provide a platform for interaction with contacts rather than restricting them to certain roles and certain responsibilities as it is the case in a traditional enterprise.

Fig.1 shows a basic picture of socially-augmented software: when a requirement is activated, the software will try to fulfil it on behalf of the corresponding user. This is compared to current software which needs certain information to decide the best way to meet the requirement [3] but not always possible to get such information through automated means. Thus, it could rely on the contacts of a person (user) for obtaining this information. Moreover, sometimes software is unable to execute all the tasks needed to enact a certain way to reach requirements or obtain the information required by these tasks. Then it would again need to rely on the users contacts. Since the selection of contact to rely on is not always straightforward, software needs to analyse the network of contacts and interact accordingly. Here we use the notion of services to denote the information provided and the tasks executed by contacts.

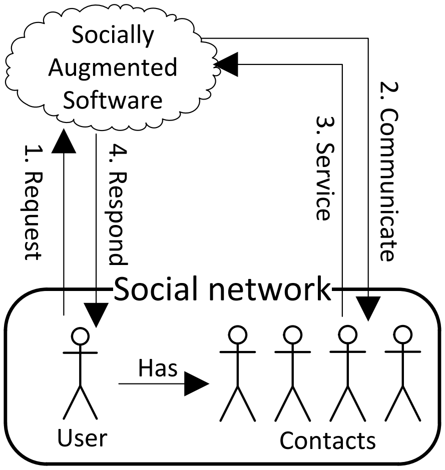

Figure 1. Socially Augmented Software Process

Social information seeking and publishing can be achieved through approaches such as social network analysis, social search, social tagging, folksonomies, social ranking, collaborative filtering, recommender systems, mashups, query log processing etc. [4], [5], [17]. These are means to achieving the 'services' that can be provided by social contacts. There is a range of engineering problems, for which the research is still in the first stages, such as the specification of socially-achievable services, the characteristics of contacts who can provide them in timely and trustworthy way, and the style a software has to follow when interacting with them. It should also note that the analysis of the state of the art utilities for socially-augmented software is outside the scope of this paper.

In our proposal, socially-augmented software is motivated by several distinguished benefits and some examples are given below:

- It enables achieving a range of requirements that would not be achievable through automated means. For example, a smart-grid system [6] aims to generate energy that is going to be ideally used. This would need to estimate the energy consumption every day which in turn requires people to explain their intended activities that are hard to capture without an individuals input.
- It enables alternative approaches for solving problems more cost-effectively in certain domains and for certain kinds of requirements and users. For example, instead of relying on a travel agency or an ICT based tour-guide solution, users could formulate groups of contacts to provide and consume desired services. The role of software in this example is to act as a platform to support systematic communications and facilitators for sharing services in a way that leads to reach requirements accurately and efficiently.

- It can avoid the deployment of expensive IT and business infrastructure. For example, in order to understand the quality of wireless network coverage in a university campus, software can be designed to allow staff and students to share information so that a user could move to an area proactively where the signal is more robust. Moreover, users could rely on each other to obtain necessary information for setting up their wireless connections. This would save much effort and costs, at both IT and business level. In this example, a wireless configurator could be designed to see online contact and communicate with them.
- It enables the formulation of socially-customized solutions within a community to which a user belongs. The solutions formulated or elected by the social contacts of a person are likely to be more acceptable by that person in comparison to those made by software developers, especially when this relates to the personal and social preferences and choices. For example, friends of a tourist would know the tourist's habits and what might attract the tourists attention in a new place. So when they suggest a route to follow, this would be influenced by that knowledge developed by observing and interacting with that tourist over time. Such knowledge would not be known by engineers and it would be hard to develop through automated learning processes as this would require capturing a large number of trips done by that user first. The dependency on social contact is one way to achieve that knowledge where there could be other techniques such as collaborative filtering and case-based reasoning. While the rest require the existence of data to reason with and infer information from, social-augmentation does not have that restriction. Perhaps the main restriction for social-augmentation is the willingness of contacts to collaborate with software.
- It enables a richer accommodation of, and adaptation to, changes. The volatility of the world and the users make it hard and almost impossible for software engineers to specify solutions which will remain valid when time passes. Design-time validation leads to results which could be valid but only temporarily as the world continues to change. Socially-augmentation boosts the awareness of software with the awareness of the social contacts in a lifelong style. For example, a fully-automated tour guide may assume that the hotels in a certain geographic area are unlikely to provide internet when this information is not explicitly said via their website or other source of information. That could be true at certain moment but may not remain like that forever. Social contacts who visited that city recently could provide that information and the software would benefit from this socially-collected piece of information and skip the step where it suggests to the tourist to buy the internet service in advance.

III. Requirements-based Conceptual Model

In this section, we use a museum-guide mobile information system [8] to illustrate the concepts introduced above and explain how socially-augmented software can be helpful in this situation. A mobile museum-guide is highly reliant on user the context and able to deliver messages and execute tasks in a context-aware manner. It also runs on portable devices for supporting visitors on the move. The common requirements of such systems include giving information about pieces of art, alerting of some events in the museum, helping on registering for services, etc. We design this system to be socially-augmented so that a user is part of a network of contacts and this network is communicable to support software in providing services to that user.

We divide the operation of socially-augmented software into three stages: (1) Requirements Activation, (2) Solutions Evaluation, and (3) Solution Execution. In the following we explain the main concepts of each stage:

A. Requirements Activation

Requirements could be activated in several ways. For example, they can be activated by a user request or a change in the environment surrounding the user (i.e. context [8]) [9]. In socially-augmented software, these requirements can also be activated by social requests through a users contacts on behalf of the user. Moreover, since the context cannot always be monitored by automated means [10], the user and their contacts could act as monitors and provide their perception of the context. Fig.2 shows the main concepts an analyst needs to activate the right set of requirements correctly. That is, the analyst needs to specify the possible ways to activate a requirement and to monitor the context. The specification should also specify which contacts can perform the activation and monitoring tasks.

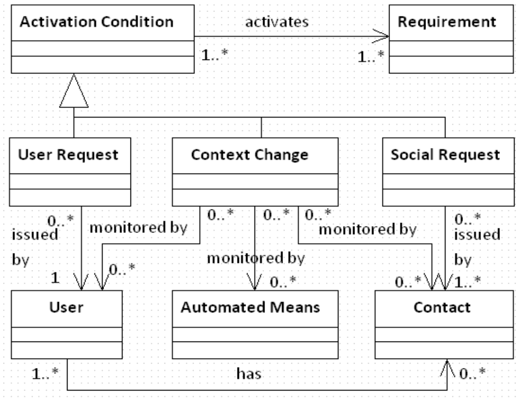

Figure 2. The main artifacts of the activation phase

Example 1. One of the requirements of a museum visitor is that the system shows information about pieces of art that could be interesting. The user can activate this requirement by following a certain dialogue. Another way is that the software monitors the context which includes the users profile, location and the nearby pieces of art, and hence activates this requirement autonomously in the right moment. This means that the right information is provided in the right place at the right time. When it is impossible to find sufficient data that describes the profile and the interest of the visitor, this information could also be obtained from social contacts. That is, the software gets all the values of relevant context attributes. When the user posts on the social network stating that they are in the museum, their contacts could further specify pieces of art which could be of interest to that user. By doing this, the contacts actually activate certain requirements for the museum guide which relates to provide information about these pieces of art when the user is a location close to one of them.

B. Solution Evaluation

Software could follow different alternative solutions in order to fulfil users requirements. Each solution has certain quality where quality is seen as a set of quality attributes covering the different facets of the degree of excellence of a solution, e.g., comfort, readability, and transparency. One way to assess a solution against a quality attribute is to rely on the user specification of what affects a quality (users beliefs). Another way is to monitor the relevant context of a user which affects the quality of each solution. A proposed way is to rely on the network of contacts of a person and how they view the quality of a solution in each of the different contexts. Similar to the requirements activation, these means are not mutually exclusive.

Fig.3 shows the basic concepts for evaluating the quality of a solution. The software has to provide ways for each user to specify their perception of quality. The analyst has to specify the context attributes influencing quality and the approach to monitor them. Finally, the analyst has to specify the contacts that are eligible and relevant to provide their quality judgments. Obviously, this met-model does not provide concepts for a complete specification; it is rather meant to provide foundation for a more fine-grained and specific concepts.

Example 2. To show information about a specific piece of art, the mobile software could follow two solutions. The first is by showing an interactive presentation and the other is to show a video. If a user perceives that interactive methods are not comfortable when using mobile devices then the software will show a video. When this information is not specified, the software could monitor the context which could, in this case, include the size of the screen and the expertise of a user interacting with mobile devices. Alternatively, the software could rely on the user contacts that have given feedback about each of these two solutions and adopt the one which is mostly fit for purpose.

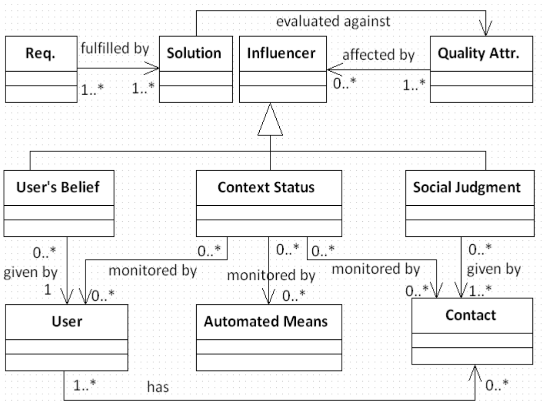

Figure 3. The main artifacts of the evaluation stage

C. Solution Execution

To comprehend a solution for achieving requirements, the software has to execute or facilitate execution of certain tasks which comprise information extraction and processing. Information could be obtained from the user, automated means or user's contacts and processed by them. However, a task could require all of the three methods together. The analyst has to specify how to execute each task and when to use each of the three methods. Fig. 4 shows the basic concepts on how to execute such solutions.

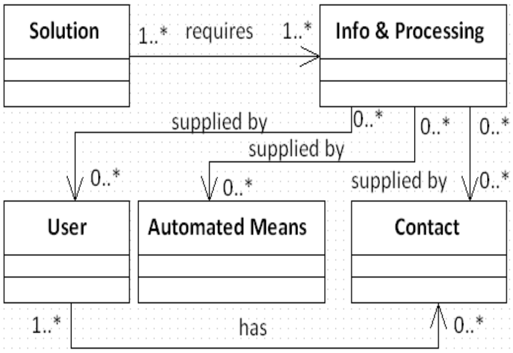

Figure 4. The main artifacts of the execution stage

Example 3. For a specific piece of art, suppose the only applicable alternative solution is to search the internet and find information regarding that particular piece. This requires the user to provide keywords about the artwork to the software and then the software will use some search engines and get a set of candidate pages. The users contacts can recommend certain pages so that the software could rely on that when prioritising the pages. Furthermore, some pages could be in a language which is unknown to the user. Some online contacts could volunteer to translate the page, and this empowers the operation. Alternatively, the software can use automated translators.

IV. Challenges for Social Augmentation

In the previous section we explained the role of contacts and positioned this within the context of different requirements modelling artefacts at various stages of the socially-augmented software process. In this section, we focus on the way users share and ask for services and specify how the software could facilitate this activity. We discuss a set of core factors that influence the interaction and interdependency of a persons contacts. That is, we discuss the baseline criteria for user to request or provide services. This is from the worldview of a service provider (information and tasks). However, this is still applicable from the perspective of a consumer.

Who. This factor stands for a set of contacts who can request and consume a service. This specification can be performed via enumerating contacts by names. Alternatively, it can be formulated as a condition on the personal profiles of contacts and their interaction styles and history. This specification introduces several challenges within the context of socially-augmented software. The main one concerns the uncertainty and incompleteness of other contacts characteristics. The provider would first formulate a mental model of each of the contacts and convey that to software. The expression of such model is not an easy task for the majority of users. Thus novel tools need to be designed. One possible way is to build it gradually so that it is incremented upon each interaction between the contact and the provider and also upon each visible interaction between the contact and other contacts. However, this introduces another challenge which is the traceability between the mental model and the changes in the characteristics of contacts over time. As a solution, we can provide tools to capture the social reputation of a contact relying on the collective judgment of the community of contacts to keep the mental model up to date.

When/Where. This factor stands for the environment that surrounds both the provider and the consumer on which they both have no control, i.e. their context. For example, a provider could offer a translation service to translate a brief description of a piece of art that is not their own. The specification of this criterion introduces the problem of monitoring both in terms of the capacity of automated monitors and privacy issues. Users might be willing to provide such services but the issue of describing their context and the related privacy concerns would make the whole process impractical and impossible.

What. This factor is about the specification of the services to be provided, i.e. the information to share and the tasks to execute. This specification faces several problems. One major problem is that when providing a service, users could be providing more than what is required. For example, sharing the information about the traffic level in a certain area conveys also that the provider is in that area. Some other information can be assembled from different services provided by a contact. Other challenges include detecting the extent to which others can infer information and execute further tasks based on the services of a contact and hence raising awareness of the risks [11]. This would probably call for privacy settings which are adaptive [12] so that contacts can collaborate with a minimized effort as they would not need to adjust and worry about consequences of disclosing more than they would like to.

Why. This factor concerns the reason for which a consumer asks for a service, i.e. the goal which is intended to be achieved. However, this imposes challenges such as traceability and guarantee of users declaration and service. To address these challenges we need to model and analyse trust relationships between users. Moreover, we need to allow another communication platform between contacts as a social monitoring facility. This is a complex problem in both real life and socially-augmented software. Recent research in digital forensics [13] could be a potential candidate to reveal whether services are used for undeclared purposes.

How. This factor stands for the way by which a contact provides a service. Two main categories comprise (i) sharing with all and (ii) sharing under demand. A person could share information like the events happening in a local area with everyone with the same interests. Sharing under demand is more appropriate for private data sharing and requires a clear request protocol from the software acting on that persons behalf. The challenge here is that contacts evolve, transform and their behaviours change with regards to the nature of information, the extended consumer contacts and the goal for which the service is going to be used. Specifying this for each piece of information and task is cumbersome. The aim is to specify the policies and then infer traceability links between the services to enable better decision making for service provision.

V. Discussion and Future Work

This paper proposes a new paradigm of augmenting a software operation with the power of the network of contacts of a user. The use of social contacts has been a part of our society since prehistoric times to provide invaluable support when achieving certain requirements. The increasing popularity of using social networks to manage and mediate social connections makes it possible to realise socially-augmented software and increases the potentials of our vision in this domain. We argue that the integration of the power of operational or business software and the power of users contacts introduces a new model for achieving requirements. This can be enabled by incorporating a users contacts to obtain information and execute tasks.

The benefits of socially-augmented software are obvious as it can significantly reduce the cost of IT and organizational infrastructure deployment. For example, a sales company could rely on customers (contacts) for monitoring the competitive technology and also the current market trends to minimize their cost. The focus within socially-augmented software is therefore to achieve a collective benefit for any individual, by ensuring contacts with common goals are connected by the most appropriate means and that they have the right tools at their disposal to make sense of the vast amount of information and knowledge that is potentially available to them.

Future work will revisit the access policies and roles privileges for organizational information systems and find out how these will be aligned with users social life and contacts. That is, socially-augmented software should not be not restricted and bounded to certain organizational rules where each user plays a specific role and has well-defined responsibilities and permissions. The challenges include the awareness of the implications of sharing, the traceability which enables a contact to ensure that the services are used for the declared reason, etc. Another challenge relates to the quality of data. Users are liable for the data quality in organizational systems; however, in socially-augmented software users are interacting on voluntary basis. Social penalties are usually informal and manifested via sanctions and decreased level of interactions and dependencies. We need to investigate the influencers on data quality to enable better design for quality.

We also need to test our entire vision on real life case studies and validate the feasibility of socially-augmented software and consequently address the issue of people acceptance of this novel software style. This makes our research multidisciplinary. Findings in sociology and psychology would be needed to augment our socio-technical paradigm. Developers tend to solve socio-technical problems with techno-social approaches. Human factors related topics [14], [15] such as human centered design, human cognition, situational awareness, information overload, training, organizational and professional culture, and trust are only some of the challenges that social-augmentation should consider.

ACKNOWLEDGMENT

The research was supported by an FP7 Marie Curie CIG grant (SOCIAD Project) and by Bournemouth University Fusion Investment Fund (BBB and the VolaComp projects).

REFERENCES

[1] F. Giunchiglia and D. Robertson. "The Social Computer: combining machine and human computation". Tech. Rep. 10-036, Trento University. 2010. http://eprints.biblio.unitn.it/archive/00001851/01/036.pdf

[2] M. Salehie, and L. Tahvildari. "Self-adaptive software: landscape and research challenges". ACM Transactions on Autonomic and Autonomic Systems, 4:2, pp. 1-42, 2009.

[3] S. Fickas, and M. S. Feather. "Requirements monitoring in dynamic environments". In the Proceedings of Second IEEE International Symposium on Requirements Engineering, pp. 140–147. IEEE, 1995.

[4] S. K. Makki, and J. Sangtani. "Data mashups and their applications in enterprises". Third IEEE International Conference on Internet and Web Applications and Services (ICIW'08). 2008.

[5] A. Rae, B. Sigurbjornsson, and R. van Zwol. "Improving tag recommendation using social networks". In Adaptivity, Personalization and Fusion of Heterogeneous Information, RIAO'10, 2010.

[6] S. Massoud Amin, and B. Wollenberg. "Toward a smart grid: power delivery for the 21st century". Power and Energy Magazine, IEEE, 3(5), 34–41.

[7] J. Krogstie, K. Lyytinen, A. L. Opdahl, B. Pernici, K. Siau, and K. Smolander. "Research Areas and Challenges for Mobile Information Systems". International Journal of Mobile Communications, 2(3) : 220–234, 2004.

[8] A. Finkelstein, and A. Savigni. "A Framework for Requirements Engineering for Context-Aware Services". In the Proceeding of the 1st International Workshop From Software Requirements to Architectures (STRAW 01), Toronto, Canada, May 2001.

[9] R. Ali, F. Dalpiaz, and P. Giorgini. A Goal-based Framework for Contextual Requirements Modeling and Analysis. Requirements Engineering, 15: 439–458, 2010.

[10] R. Ali, C. Solis, M. Salehie, I. Omoronyia, B. Nuseibeh and W. Maalej. Social Sensing: When Users Become Monitors, In the Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering, ESEC/FSE'11, 2011.

[11] I. Omoronyia, L. Cavallaro, M. Salehie, L. Pasquale, and B. Nuseibeh. Engineering adaptive privacy: on the role of privacy awareness requirements. In the proceedings of the 35th International Conference on Software Engineering (ICSE'13), 2013.

[12] M. Langheinrich. "A Privacy Awareness System for Ubiquitous Computing Environments". In the Proceedings of the 4th international conference on Ubiquitous Computing (UbiComp'02). 2002.

[13] E. Casey. Digital Evidence and Computer Crime Forensic Science, Computers and the Internet (3rd Ed.), Elsevier, 2011.

[14] ISO 9241-210, Human-Centred Design for Interactive Systems, 2010.

[15] H. Dogan, S. Ahlberg Pilfold, and M.J.d. Henshaw. The role of Human Factors in addressing Systems of Systems complexity. In the Proceedings of the IEEE Systems Man and Cybernetics (SMC'11), 2011.

[16] A. K. Chopra and M. P. Singh. "Colaba: Collaborative design of cross-organizational processes". In the Proceedings of the Workshop on Requirements Engineering for Systems, Services and Systems-of-Systems (RESS), 2011.

[17] M.A. Kabir, J. Han, and A. Colman. "Modeling and Coordinating Social Interactions in Pervasive Environments". In the Proceedings of the 16th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS), 2011.