2015 IEEE Trustcom/BigDataSE/ISPA

Trust-aware Elastic Social Compute Units

Mirela Riveni, Hong-Linh Truong, Schahram Dustdar
Distributed Systems Group, TU Wien
Email:{m.riveni,truong,dustdar}@dsg.tuwien.ac.at

***Abstract*—With the advance of research in human computation, applications and software systems are increasingly being designed to include the human aspect of computation. We work with Social Compute Units (SCUs) that are computational constructs with people as their core resources who use software services to organize and conduct their work. SCUs are collaborative units, and have a cloud-like behavior in the sense that they are elastically provisioned and adapted at runtime. Systems that utilize the concept of SCUs bring challenges that are associated with the highly dynamic and unpredictable human-centric behavior. Thus, trust in human based services is of paramount importance. While there is related work on social trust in the social networking and crowdsourcing areas, trust in highly coordinated constructs such as SCUs remains a significant challenge. In this paper we provide a trust model that considers merging social trust with performance-based trust of human based services into an integrated trust model for SCUs. We illustrate the models' application in concrete strategies, such as for elastic management of SCUs and incentives for SCU members.**

I. Introduction

Complex collective adaptive systems (CASs) that are on the rise in the recent years, offer new ways of resource utilization and promise novel complex applications that ease and smarten the way societies function [17]. However, they also create new challenges in dealing with the plethora of resource types and services that they are comprised of. Social Compute Units (SCUs) [4] represent a form of collective adaptive systems that consist of services provided by humans as compute resources. A crucial factor for the existence of SCUs are today's human context-based profile and performance managing platforms, from which people can be recruited to be utilized as compute resources. By this we mean platforms, such as crowdsourcing ones, social-networks, expert-network platforms, and enterprise pools of human resource profiles. Moreover, as already discussed in [5], today it is feasible to model and utilize human-based task execution under the service oriented model via human-based services, and to manage these services in a programmatic way.

SCUs are formed to reach a certain task-oriented goal, with a specific quality and in a specific domain. In our work and throughout this paper, we use the term Individual Compute Units (ICU) for members of an SCU, i.e., people that can expose their skills and the results of their work as services. The purpose of the SCU or the domain of the task to be executed, is what gives context to the social relationships of its members. We work with SCUs because they are designed as solutions for applications and processes that require executing highly-complex tasks that can be accurately solved only with human-in-the-loop approach. As opposed to simple crowdsourcing tasks that require individual work, SCUs address complex tasks that may be interdependent and that require highly coordinated team-work that needs to be automatically managed, with high responsiveness to events during runtime such that the performance and quality of task results can be kept at their required level. SCUs are formed on customer request with specific requirements, and they dissolve when all tasks are successfully executed. Thus, they bring challenges that are associated with the highly dynamic and unpredictable human-centric behavior. Trust is one of these challenges. Selecting not only the appropriate members according to specific requirements but also the best among the available ones is important for forming an effective SCU.

For social networks and crowdsourcing platforms it is already identified that trust is important in deciding with whom to establish connections and with whom to interact [8], [22], [2]. We stress that this importance is even higher in SCUs because they entail complex, structures and organized work. In a previous work [15] we have argued that ICUs should be managed *elastically* for optimal SCU performance and customer cost savings. This implies that ICUs can be added and removed from the SCU at runtime based on different strategies, so that the SCU capabilities and performance are optimized at any time and for any changes in customer requirements. Hence, as much as trust is needed at the stage of the formation or selection of SCUs, it also plays a crucial role during their lifecycle for runtime adaptation, i.e., for *managing* SCU execution and thus *controlling* the level of non-functional parameters and quality of the returned results.

As far as we know, online social trust in the context of Human Computation [14] is only explicitly investigated in the context of social networks such as in [22] and [8], the Crowdsourcing context [2], as well as in human-enhanced service oriented environments on individual worker-basis [16]. However, there is no trust investigation based on metrics relevant for task-executing *elastic collectives* of human based services such as SCUs, although there is some work on trust in virtual teams (e.g., [9]). Nevertheless, SCUs are much more complex formations than those of online teams. In terms of their controlling, coordination, communication, task execution and management, SCUs can be composed and controlled automatically. To be able to utilize trust as an indicator of efficient collaborations, in this paper we propose a trust model for SCUs as well as for their individual members, we analyze how metrics used within elastic algorithms for adaptation of SCUs affect member trust within SCUs and in the general network to which the member belongs, and make a correlation between member-trust within SCUs on one side and the trust on SCUs as a whole on the other. Thus, the key contributions of this work are:

- a socio-technical trust model for SCUs
- elasticity algorithm with trust-updates
- a novel incentive strategy for ICUs (SCU members), based on our trust model

The remainder of this paper is organized as follows. In Section II we give further motivation for our work and continue

978-1-4673-7952-6/15 $31.00 © 2015 IEEE
DOI 10.1109/Trustcom-BigDataSe-ISPA.2015.367

with preliminaries for our model in Section III. In Section IV we present our model of SCU trust. In Section V we describe the application of the model in different strategies, while in Section VI we evaluate a proposed trust-based elastic adaptation strategy. We discuss related work in Section VII, and conclude the paper in Section VIII.

II. SCU Fundamentals, Motivation and Challenges

A. SCU Fundamentals

SCUs are virtual units of multiple expert human-based resources that use software services for executing complex tasks. They are formed on customers' request and with customer-set constraints, such as those for the skills and price of the human resources for specific tasks. However, they can also be formed voluntarily and in an ad-hoc way for a specific purpose. Thus, the nature of SCUs is task-oriented for completing a goal, where tasks might come from customers but also generated by the SCU members during SCU execution, i.e., at runtime. In addition, an SCU is elastic but it is so as much as the platform provisioning it allows. In [15] we have discussed SCUs runtime execution, its elastic properties and presented elasticity management strategies that SCU provisioning platforms should support. Hence, utilizing the concept of SCU, as a construct that gives the possibility to approach the issue of human computation in terms of collectiveness [13], we can argue the importance of collective trust in the success of programmable[1] SCUs.

B. Illustrative Scenario-Predictive Maintenance

We demonstrate SCU related concepts and argument the importance of SCU trust with a scenario of *predictive maintenance*. Let us take the case of chiller maintenance in buildings maintained by an enterprise. In M2M (Machine-to-Machine) environments that are supported by Cloud provider infrastructures and platforms (IaaS and PaaS), these types of chillers are monitored by different types of software and devices (e.g., sensors), while the monitoring data is analyzed on-site or remotely. Operations enterprises that provide remote monitoring services typically monitor the working of chillers and report to experts of any anomalies, who then try to figure out the dysfunctional component and try to fix it according to predefined procedures. Thus, the typical maintenance is based on incident management that is not predictive. Furthermore, this process sometimes is counter-effective because it can happen that procedures themselves that are taken for maintaining or increasing chillers efficiency cause problems. Thus, the experts cannot make informed conclusions as to if the problem was related to the undertaken maintenance procedure, or it was a fault in one of the chillers components. What is needed is predictive maintenance so that malfunctions are lessened and the very high costs of chiller replacement and/or their operation costs (e.g., high electricity costs) are avoided. In today's direction of management towards smart buildings and smart cities, this problem becomes even a greater challenge as a consequence of *Big Data*. These three challenges, namely deploying predictive processes, better cause analysis and *Big Data* management demonstrate the need of having multiple SCUs with expert members of different areas. Consequently, we face the challenge of which expert capabilities to trust and engage in SCU task execution. In this scenario multiple SCUs may be formed as follows:

a) SCU with ICUs that have *data science* expertise who will manage *Big Data* coming from monitoring devices about the different monitorable and measurable chiller properties and components, such as: the temperature of the water going in and out, the compressor state, the evaporator state, the water flow, the state of the condenser, the on/off status of the chiller but also environmental data. This SCU collects the data, cleans it, filters it, structures it and provides it in the form of Data as a Service (DaaS).

b) SCU with expertise in chillers and IT, that conducts *data analysis* to detect malfunction probability in the chiller or problems that are connected with the environment, (e.g, improper water used that causes problems in the tubes).

c) SCU with on-site *technical* experts that gets directions from the analytic SCU (in b)) and takes action as required, e.g, adapts the parameters if and when needed, and takes care about chiller components. These different SCUs collaborate through a common platform.

Cities (i.e, city governance bodies or contracted companies) are the clients of SCU provisioning enterprises in this case (those that provision the collaboration and communication platform). Clients are given two options for choosing the SCUs that will execute their tasks, they can chose to use existing available and trusted SCUs that have previously worked on similar tasks, or to request new SCUs to be formed. Figure 1 depicts the environment within which an SCU works, and crucial operations for an SCU provisioning platform that manages SCUs. The SCU provisioning platform keeps a record of registered ICU profiles, including their expertise, and logs their information regarding the SCUs in which they have been included (so it maintains SCU profiles as well). These ICU profiles may be registered and hosted on the SCU provisioning platform but they can also be references to other online resource pools such as crowdsourcing platforms, expert networks and social networks. The platform can run SCU formation algorithms to create new SCUs from trusted ICUs, or it can run a ranking algorithm for whole SCUs to chose the most trusted one, depending on the submitted client request. The selected SCU, or the newly formed one, is monitored at runtime and may be elastically adapted in response to different monitoring events and based on different elastic strategies to best fit customer requirements. The adaptations include activities, such as task reassignments, delegations, addition of new ICUs and removal of existing ICUs. Thus, the capabilities and expertise of ICUs can be adapted, and consequently, the performance of the SCU can be changed at runtime. The SCU collaboration is supported by Communication Services, Cloud Services, and also data from Internet of Things devices. The communication is conducted through a middleware. A detailed description of an SCU provisioning platform is out of the scope of this paper.

C. Observations and Challenges

From the described environment we derive the fundamental trust related observations in SCU provisioning, generalized across domains:

- Because an SCU is based on expertise and task execution, a trust (and reputation) score of an ICU to be included in the SCU should not be calculated solely from the social trust scores of collaborators out of interaction satisfaction, rather its performance within

[1] By programmable SCUs we mean SCUs that are automatically manageable as for example software services and compositions.

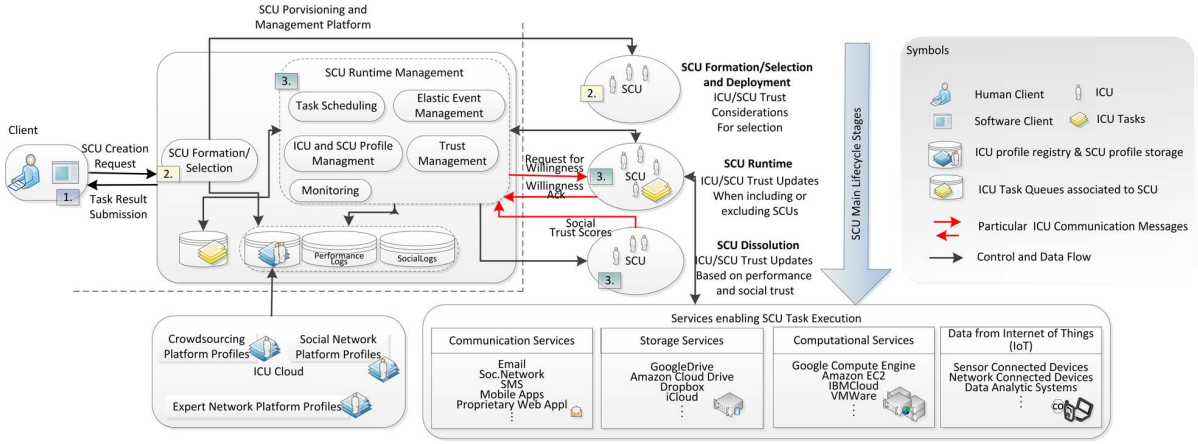


Fig. 1: SCU - Conceptual Platform and Working Environment

the required expertise should be considered as well.

- SCUs are elastic in term of structure, price and non-functional parameters. Consequently, the SCU trust is elastic as well.
- SCU is a socio-technical formation that has ICUs (people who offer their capabilities as services) as its core resources, thus its nature is unpredictable and trust can play a crucial role in regulating their behavior.

These observations bring us to the following corresponding research challenges:

- What are the fundamental metrics to be included in a socio-technical trust model that will include both the social and performance contexts of ICU?
- How is trust updated and how it fluctuates within the dynamics of an elastic SCUs lifecycle?
- How can trust be included in elasticity strategies for effective SCUs? What can be a feasible trust-based incentive strategy for ICUs?

Because we have identified that trust is an indicator of key importance in SCUs, in this paper we present a trust model for SCUs including trust-update and incentive mechanisms, through which we address the aforementioned challenges.

III. Preliminaries

A. ICU and SCU Notation

To come to our trust model we extend some work, in particular work on metrics, that we have previously presented in [15]. In line with our previous work, we use here the same annotation for a cloud of ICUs, which is our universal set of ICUs as $R = \{r_1, r_2, r_3...r_n\}$, whereas we denote the set of ICUs that are members of a particular SCU as $S = \{s_1, s_2, s_3...s_n\}$, where $S \subset R$. We denote the collection of SCUs in which an ICU has been a member of, over a specific time period with $\mathfrak{U}^{\tau} = \{S_1^{\tau}, S_2^{\tau}...S_n^{\tau}\}$.

B. Context

To account for the possibility of multiple capabilities of people, it is important to note here that our metrics for calculating the Socio-Technical Trust for ICU and SCUs are calculated in the context of a particular skill or expertise of an ICU, for which an ICU is invoked in the SCU; e.g., a data scientist can conduct tasks related to structuring and filtering data to provide DaaS but he/she can also be involved in another SCU for analysing data and creating operations plans. Consequently, in fact we have separate roles for the same ICU, so the same ICU is treated as separate in the context of its expertise within different roles (icu1 in Figure 2). Tasks are assigned based on a match of the skill needed to execute the task and the skill that an ICU possesses. We define the set of skills of an ICU with $SK(s_i) = \{sk_1, sk_2, ...sk_n\}$, and the set of tasks assigned to an ICU with $T(s_i) = \{t_1, t_2, ...t_n\}$, where $\forall t \in T, \exists sk \in SK$. The set denoting the skill-task/s pairs is $\mathfrak{S}\mathfrak{T} = \{ST_1, ST_2, ...ST_n\}$, with its elements defined as $ST = \{(sk_1, \{T_{sub}\}) \mid T_{sub} \subseteq T \wedge T_{sub} \neq \varnothing \wedge sk \in SK\}$. The context C, in which we calculate a metric M for an ICU corresponds to the skill type with which the ICU is invoked in an SCU, so we have $C(M, s_i) = sk \Leftrightarrow sk \in ST$.

C. Metrics

Based on our previous work on ICU performance as well as on some related work, we have identified key performance indicators for ICUs and SCUs that we use in ICU/SCU elastic management. We discuss some of them in this section, as they are part of the core of our socio-technical trust model.

Effort - the effort of an ICU is the average time spent by an ICU for executing a task, thus it indicates timeliness. The SCU effort is an aggregate of the effort of its members-ICUs.

Productivity - we define ICU productivity as the number of successfully executed tasks over a time unit. Thus, productivity is the ratio of successfully executed tasks that are submitted by an ICU as a result, and the total number of tasks executed per time unit. The SCU productivity is an aggregate of the productivity of the constituting ICUs.

Reliability - we derive ICU reliability from multiple atomic metrics. Namely, in our previous work we have presented two novel metrics, Willingness and Willingness Confidence-Score. To clarify these metrics, we note that previously we have introduced a task delegation mechanism that would avoid overloaded ICUs [15]. The mechanism achieves this by asking appropriate and available ICUs if they are willing to execute a particular task that needs to be delegated and reassigned, and accepting acknowledgments from ICUs if they are willing to execute that task. Thus, we presented a metric that we call Willingness, and defined it as the ratio of the number of tasks that are assigned to an ICU in response to its willingness acknowledgments, and total willingness requests that are sent to an ICU for working on tasks that need to be reassigned.

TABLE I: Notations

Notation	Description
s_i	ICU, a member of an SCU
scu_i/S	SCU/an SCU as a set of ICUs
r_i	ICU not associated to any specific SCU, a global profile of an ICU in the pool of ICUs
m	Number of invocations of a particular SCU
$mc(s_k, s_i)$	Trust vote from ICU k to ICU i based on their collaboration experience within the same SCU
$MCTS(s_i), MCTSgl$	Local, respectively, global Membership Collaboration Trust Score of an ICU,
$STTR$	Socio-Technical Trust score given to ICUs based on the STT of the SCUs in which it has been a member
$PT(s_i)$	Performance/Technical trust of an ICU in SCU
$ST(scu_i)$	Social Trust of a specific SCU
STT^f	Socio-Technical Trust metric of an SCU at the time of formation
STT^e	Socio-Technical Trust metric of an SCU that has been invoked before

Moreover, we inferred another novel metric, the Willingness Confidence-Score of an ICU which we defined as a product of Willingness and the rate of success in executing the number of tasks that are delegated to the ICU as a result of its own feedback for willingness to execute those tasks,

$$WillingnessConf = \frac{SentAcks}{ReceivedReqs} \times \frac{DelegatedTasksExecuted}{TotalDelegatedTasks} \text{ [15].} \quad (1)$$

Thus, the Willingness Confidence-Score is a measure of the *delegation reliability* of an ICU for tasks that have been delegated to it, because it shows how true to its own statements an ICU is by accounting how many tasks it has executed of those it claimed that it is going to execute. To come to a comprehensive reliability metric for an ICU we combine the *delegation reliability* with the total successfully executed tasks of an ICU, including tasks that are initially assigned to it,

$$Reliability = WillingnessConf \times \frac{NonDelegatedTasksExecuted}{TotalNonDelegatedTasks}. \quad (2)$$

Quality of Results-Client Satisfaction - the quality of tasks results (QoR) depends on multiple factors, such as the domain of the SCU goal, the task types, the SCUs structure [18]. In this work we define it as the client satisfaction from the task results. Thus, in our model this metric is calculated as a vote from the client to the SCU. However, QoR can be set up to be automatically measured in some cases when expected task results are specified, so it can be included in our model when its calculation manner is defined (see Figure 3). In order to ease the flow of the discussion that follows, Table I gives some of the frequently used notations used in our model.

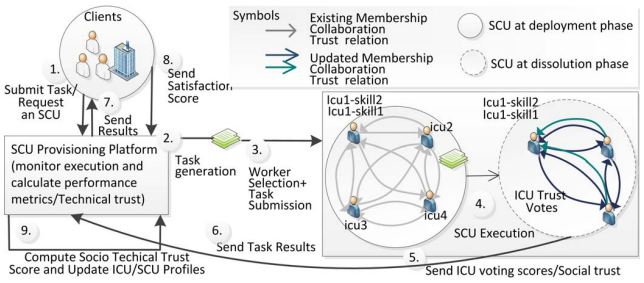


Fig. 2: SCU operation and ICU trust updates

IV. SCU Trust Model

A. ICU Socio-Technical Trust and Reputation Model

To come to an SCU trust model, we first define ICU trust. ICU trust can be seen from two perspectives: a) that from an SCU in which it is a member of, and b) from the global perspective of its performance and social impression derived from each of its engagements in task execution within all SCUs of which it has been a member of. Because the latter implies a global trust score for an ICU, in this work we call it reputation of the ICU, to differentiate it from its local trust score within a specific SCU. We define an ICUs software-based trust, called performance trust as the product of its productivity and reliability, which are calculated from automated monitoring.

$$PT(s_i) = Productivity(s_i) * Reliability(s_i) \quad (3)$$

To come to a social trust of an ICU in the context of an SCU we define the Membership Collaboration Trust Score of an ICU within the specific SCU. For this metric we propose the strategy that each ICU within the same SCU votes for all other ICUs of the same SCU before the SCU is dissolved. This vote is given for the interaction satisfaction with the ICUs with which the voting ICU has interacted within the same SCU or, in the case if it has not interacted with, the vote is cast based on the perception of what the ICU to whom it casts a vote has contributed for the SCU. Thus, we assume here that every member has either interacted with all others at least once or if not then at least it knows what all members have contributed. Consequently, the Membership Collaboration Trust Score of an ICU is calculated from the votes that each member of the SCU gives to another member of the SCU before the SCU is dissolved as in Equation 1. This metric is a weighted aggregation of scores, such that the votes of ICUs with higher reputation are given higher weight. In this work, in all equations the sum of the weights is 1, and every metric value is in the range of (0,..1].

$$MCTS(s_i) = \sum_{k=1,k\neq i}^{|S|} w_(s_k) * mc(s_k, s_i)/|S| - 1 \quad (4)$$

From the global perspective, the Global Membership Collaboration Trust Score of an ICU in regard with all SCUs that it has been a member of, is an aggregate of its MCTS score from every SCU of which it has been a part of, taking into account the number of invocations of each SCU, because an SCU can be invoked multiple times for the same or different clients. In addition, because human behavior is highly unpredictable and changes with time, the inclusion of a time restriction is important. Thus, a time period limit for the number of SCU invocations, has to be assigned in calculations, with the purpose of accounting for the freshness of trusted relations.

$$MCTSgl(s_i, \tau) = \frac{1}{|\mathfrak{U}^\tau|} \sum_{j=1}^{|\mathfrak{U}^\tau|} \frac{1}{m} \sum_{l=1}^{m} MCTS(s_i)_{l,j}, \quad (5)$$

where m is the number of invocations of a specific SCU, of which an ICU has been a member, and $\mathfrak{U}^\tau$ is the total number of different SCUs that the ICU has been a member of, within a specific time τ. The local Socio-Technical Trust of an ICU (in the context of a specific SCU invocation) $STT(s_i)$ is a weighted sum of its performance trust and the Membership Collaboration Trust Score from its co-members in the SCU.

$$STT(s_i) = w_{pt} * PT(s_i) + w_{mcts} * MCTS(s_i) \quad (6)$$

We name the global Socio-Technical Trust Score of an ICU, $Reputation(s_i)$. This is in line with the concept that trust in an ICU is individual, whereas reputation is a global metric that includes the trust scores from all the actors in the system who have interacted with the ICU. We calculate the Socio-Technical Trust of an ICU as a sum of its STT and a Socio-Technical Trust score that we assign to it based on the STT scores of the SCUs to which the ICU have belonged, considering all invocations of the ICU within different SCUs in a specific period of time.

$$Reputation(r_i, \tau) = \frac{1}{|\mathfrak{U}^\tau|} \sum_{j=1}^{|\mathfrak{U}^\tau|} \frac{1}{m} \sum_{l=1}^{m} (STT(s_i)_{l,j} + STTR(s_i)_{l,j}). \quad (7)$$

B. SCU Socio-Technical Trust Model

We consider two cases where trust is important for SCUs, namely a) when an SCU is newly-formed and composed from ICUs with appropriate expertise for the SCU tasks, and b) when the client may chose an already existing SCU. For SCU formation algorithms, we propose an aggregated trust score that is based on the reputation score of ICUs over which the selection algorithm is executed. As every skill does not have the same importance within an SCU, weights are given to ICUs for different type of skills. For example in our scenario, data scientist members may have higher weight than ICUs with other roles. In different domains, the importance of expertise may vary drastically depending on the SCUs goal. The Socio-Technical Trust for an SCU that is newly formed is calculated as a weighted aggregate score of the reputation of the ICUs of which it will be formed, because the reputation contains the performance as well as the social trust of ICUs. Thus, this metric can be used in formation algorithms to decide about the most trusted SCU from the possible compositons.

$$STT^f(scu_i) = \sum_{i=1}^{|S|} (w_{expertise}(s_i) * Reputation(s_i)) / \sum_{i=1}^{|S|} w_{expertise}(s_i) \quad (8)$$

In the case when a customer or a software client wants to chose an SCU that has been previously invoked, our model takes into account SCU metrics. SCU specific metrics are aggregates of the ICUs that have taken part in it and it is also a function of performance-based trust (Equation 9), and social trust (Equation 10). PT or software-based trust of an SCU, which is the *technical* part of trust in our model, is an aggregate of the performance trust of its member ICUs, considering the total number of SCU invocations over a period of time, m.

$$PT(scu_i) = \frac{1}{m} \sum_{k=1}^{m} \sum_{i=1}^{|S|} PT(s_i)_k / S \quad (9)$$

$$ST(scu_i) = \frac{1}{m} \sum_{k=1}^{m} w_{mcts} * MCTS(scu_i)_k + w_{css} * CSS(scu_i)_k. \quad (10)$$

The social trust of an SCU, STT, is calculated as a weighted average of MCTS of its members, and the customer satisfaction score of the client (CSS) regarding SCUs performance and quality of results. Here too, CSS is a subjective trust of the client toward the SCU. Equation 11 defines the Socio-Technical Trust of an SCU that can be used in SCU selection algorithms, because it considers historical data.

$$STT^e(scu_i) = \frac{1}{2}(PT(scu_i) + ST(scu_i)) \quad (11)$$

Figure 3 shows the metrics that we have discussed and gives an overview of how we model the Socio-Technical Trust Score of an SCU. The arrowed lines are in the direction from more simple metrics, to metrics which are comprised of those simples ones, and show how we derive the STT for SCUs from both the social scores and automated scores based on SCUs performance monitoring. As QoR metric is domain-dependent we have not included it in our definitions but the model shows that it is easy to add this and any other metric to it.

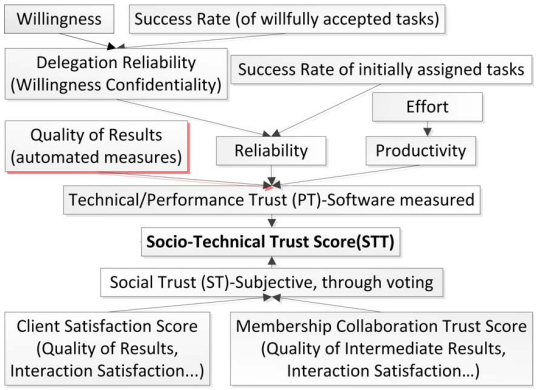


Fig. 3: SCU Socio-Technical Trust Model Metrics

V. Utilization of the SCU Trust Model

A. A Trust-based Incentive Strategy for SCU Members

Social Capital as a collective gain is seen as one of the main drivers for people to cooperate [20] and perform well in collaborations. We were motivated by this concept to model $Reputation(s_i)$ but most importantly to use it as an incentive method to control the behavior of SCU members for performing efficiently and avoid misbehaviors. In calculating the reputation of an ICU we include the Socio-Technical Trust scores of each SCU that it has been a member of. Equation (7) and Algorithm 1 describe the reputation update that is our incentive mechanism. With this mechanism, ICUs become aware that the total trust score of the SCU in which they are engaged, will affect their reputation and so this may influence their motivation to work and the SCU provisioning platform can avoid or lessen misbehavings within the SCU. This incentive mechanism is enforced with designing the ICU/SCU supporting platform in a way that gives ICUs the knowledge that their performance data and the Membership Collaboration Trust Score are included in the SCU STT score, and in turn the SCU STT score is used to update their reputation. Lines 2-7 in Algorithm 1 check if all ICUs have finished their tasks and ask them to give a membership collaboration trust score to every other ICU in the SCU. Lines 8-14 calculate ICU and SCU STT scores, according to the presented model.

In [16] the authors argue that people have low incentives to manually assign ratings. To overcome this challenge, our strategy enforces Membership Collaboration Trust score voting as a condition for an ICU so that its work can be accepted and a payment can be made before the SCU is dissolved. Thus, lines 15-18 update ICUs' Reputation, pay the ICUs *after they have voted*, and the SCU dissolves. On the other hand, the incentive mechanism obviously brings the challenge of keeping ICUs from misbehaving while voting, because the Socio-Technical Trust score of the SCU in our model is higher the higher the trust score between SCU members is, thus ICUs can be tempted to give a higher $mc(s_k, s_i)$ to increase the SCU trust so that their reputation is also increased. This might lead to

unfairness, because ICUs can vote high for others only for their own gain. To overcome the issue on unfair ratings, based on the voter ICUs current reputation, we give it an allowed trust range for voting (line 5 of Algorithm 1). This range can vary according to the range values that a user will impose on the model. In this work, if the voter ICUs (sk) reputation is lower than 0.5, we chose the allowed voting ranges for voted ICUs (si) as follows:

$$mc(sk, si) = \begin{cases} [0.5, 1] & if \quad Reputation(si) \geqslant 0.5; \\ [0.1, 0.5] & if \quad Reputation(si) < 0.5. \end{cases}$$

B. A Trust-based and Cost-effective SCU Elasticity Strategy

The objective of an SCU provisioning platform is not only to provision an appropriate SCU for a specific client, but also to maximize performance, which means keeping deadlines, submitting high quality results and maintaining the SCU operation cost within the allowed budget. In our scenario, e.g., we need to ensure that DaaS is always available for the second SCU(Section II/B/b), and that a person is always available for immediate response for the third SCU (II/B/c). This requires for ICU availability at all times for all SCUs. For this, to optimize performance with Algorithm 2 we propose an approach for elastic SCU adaptation considering ICU trust (that includes availability and performance metrics). The algorithm monitors each task and in case of a time-threshold at one ICU, it delegates the task to another available ICU. Lines 2-6 check for the reputation score and current execution state of the ICU from which the task needs to be withdrawn and delegated to another. If its trust score is less than 0.5 and it does not have tasks in execution it is removed from the SCU. Lines 8-12 check ICU reputation and cost for all ICUs in the pool of available ICUs, regardless if they are members of the SCU or not. ICUs that have a reputation higher or equal to 0.5 and that fit the allowed cost are ranked in ascending order of their task queues. A request to work on the task that needs to be delegated is then sent to the ranked list of ICUs. The task is finally reassigned to the ICU highest on the ranked list that has sent an acknowledgment for willingness to execute the task. If the ICU is not a member of the SCU, it is added. The SCU trust is updated at runtime during elastic adaptation of the SCU.

Algorithm 1 Membership-Collaboration Trust Update Algorithm as an Incentive Mechanism

Require: icu worker in SCU
1: **for all** icu in SCU **do**
2: **if** $icu.getStatus() == FINISHED\_TASKS$ **then**
3: $icuCurrent = icu.geticuId()$
4: **for all** icu in SCU && icuId! =icuCurrent **do**
5: getTrustRange(icu)
6: votes $\leftarrow mc(icuCurrent, icu)$
7: icuCurrent.UpdateMCTS(votes)
8: $icuMCTSList \leftarrow getMCTS(icuCurrent)$
9: $icuPTList \leftarrow getPT(icuCurrent)$
10: /* calculate SCU Socio-Technical Trust */
11: $scuMCTS = calculateSCUMCTS()$
12: $scuCSS = getClientSatisfaction()$
13: $scuPT = calculateSCUPT()$
14: $scuSTT = calculateSCUSTT()$
15: **for all** icu in SCU **do**
16: /* calculate the reputation of ICUs by adding the SST score derived from the current SCU, to the existing global Socio-Technical Trust score */
17: $icu.Reputation = UpdateReputation(icu, scuSTT)$
18: $icu.Payment = getICUPayment()$
19: $SCU \leftarrow SCU \setminus ICU$ /* dissolve SCU */

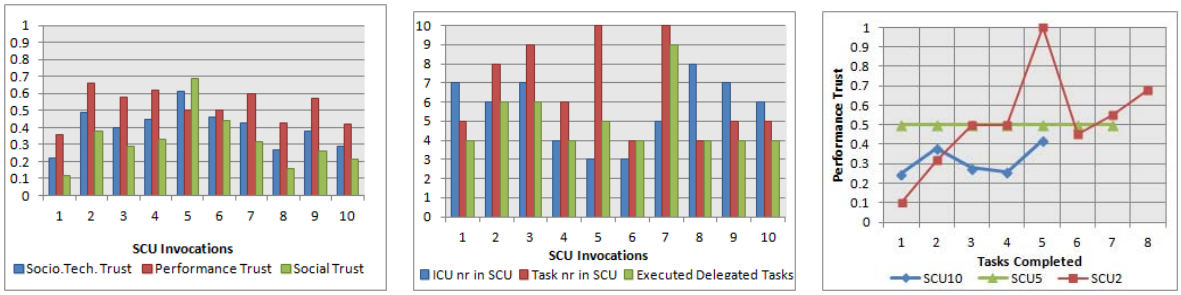
Algorithm 2 A Cost-Effective Algorithm for Elastic Adaptation of SCUs based on ICU Reputation

Require: icu member of SCU, icu member of P
Require: task assigned in SCU
1: **for all** task in T **do**
2: **if** $task.inIcuQueueDuration ==$
3: $task.thresholdDuration$ **then**
4: $icuCurrent = task.getICU()$
5: **if** $icuCurrent.icuReputation \leqslant 0.5$ &&
6: $icuCurrent.taskExecuting = 0$ **then**
7: SCU $\leftarrow$ removeICU()
8: **for all** icu in P **do**
9: **if** $icu.STT \geqslant 0.5$&&
10: $(currentCost + icu.Cost) \leq Budget$ **then**
11: $rankingList \leftarrow min(icu.taskQueue)$
12: sendWillingnessReq()
13: /* Assign the task on threshold to the highest ranked ICU that acknowledges a Willingness request */
14: **for all** icu in rankingList **do**
15: **if** $icu.Ack() == true$ **then**
16: $icuCurrent = icu$
17: **if** icu is not a member of SCU **then**
18: $SCU \leftarrow addICU(icuCurrent)$
19: **break**

VI. Experiments

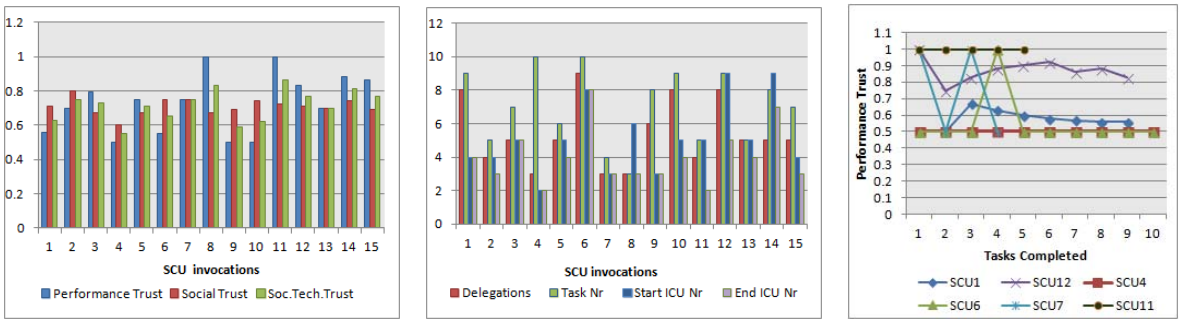
We evaluated our trust model via simulations. We designed ICU Profiles with static properties (unaltered values) and dynamic properties (updated at runtime). The static properties are skill type and cost per task, whereas the dynamic properties are: Productivity, MBCT, ICUReputation, STT and all the atomic metrics discussed in this work, all of which are calculated according to our model. Nevertheless, we note here that we calculate social trust only with membership collaboration trust scores not including the client satisfaction. Tasks are designed with states: ASSIGNED, INEXECUTION, SUCCESS, FAILURE (if in FAILURE state a task is delegated). Tasks are individually executed and they are not interdependent.

Base Algorithm We implemented an algorithm which we take as a base for comparison and which adapts the SCU without considering trust. We assigned tasks randomly and delegated randomly to ICUs. Graph a) in Figure 4 shows the STT score, MCTS and Productivity for multiple invocations of an SCU. We calculated trust assigning different weights to performance and social trust (0.4 for PT and 0.6 for ST). Figure 4(c) shows performance trust updates of particular SCU invocations at runtime. If we take the 10th invocation as an example, we can see that we have four delegated tasks out of five in total so the decline in the performance trust (productivity) in Figure 4(c) comes at the point when delegations occur. If we look at Figure 4(a) and Figure 4(b) for SCU7, we notice that the performance trust is higher than the social trust, because here we have a high number of delegations and exclusions of ICUs. This means, that a small number of ICUs executed high number of tasks but the social trust of the SCU is low because many ICUs failed to execute tasks.



(a) (b) (c)

Fig. 4: SCU metric updates within time



(a) (b) (c)

Fig. 5: SCU metric updates within time

Algorithm 1 In order to evaluate our trust models behavior we implemented the aforementioned Algorithm 1 with the following settings: we set all the ICUs' reputation in the SCU to be higher than 0.5 and thus we assigned the delegated tasks only to ICUs that have MBCT higher than 0.5. We set the weight of both the social and performance trust of the SCU to be the same, i.e. 0.5. We set ICUs that had previously failed at all assigned tasks to be excluded from the SCU after they delegate a task. Analyzing the results of this experiment we see SCU adaptations due to delegations. In some SCU invocations tasks are delegated to SCU members, in some they are delegated to ICUs that were not members and so there were inclusions of new ICUs and exclusions of ICUs. Figure 5 shows the end results of each SCU invocation (out of 15), while the graph in Figure 5(c) shows performance trust updates of some selected SCUs at runtime. Examining the case of the SCU11 from Figure5(a) we noticed that its PT was 1 while ST was lower and the STT was high as well. From Figure 5(b) we noticed that the number of ICUs is lower at SCU dissolution than at start, which means that some ICUs are excluded and thus their productivity is not included in the PT calculation. Hence, the PT is constant at 1.0. Examining ICU12 we saw that at runtime the number of ICUs also changed with delegations. Some tasks were delegated to ICUs within the SCU, whereas some new ICUs were included for executing a task that needed to be delegated. An ICU with a failed task was not excluded from the SCU and its performance was included in calculating the SCUs PT. Hence the lower points in PT Figure 5(c) for SCU12. In all cases, those ICUs that had previously performed well but had failed in a task or two were still included in the STT calculation so the variations of the trust score to lower values is also due to this fact and not only due to the number of failed tasks. For excluded ICUs the MBCT is low, in our implementation this value is excluded from the ST calculation at runtime. Hence the social trust values are relatively high in Figure 5(a).

Comparing the two algorithms it is clear that the algorithm implemented with our STT model keeps a steady high-value trust of an SCU during runtime as well as with time brings a steady performance of an SCU as compared to a strategy that adapts the SCU without considering trust (both social and technical). As much as experimenting with real data sets would give more accurate results it is very difficult to conduct the experiments due to lack of available data sets that fit the SCU construct. We designed the ICU profiles the best we could to simulate human profiles and behavior. The experiment also reflects the advantage of using our model in adaptation strategies for SCUs, because based on our model the SCU can be tuned in terms of structure and non-functional parameters to keep a certain level of desired performance.

VII. Related Work

Social Compute Units The fundamental work that first introduced Social Compute Units is presented in [4]. SCUs utilization in dynamic processes is presented in a recent work [6]. The concepts and proposals for provisioning human capabilities within the service oriented model are presented in [5]. In our previous work, [15], we introduced a conceptual architecture for a platform that provisions elastic SCUs, along with important metrics and strategies that can be used to manage SCUs in an elastic manner. This work is a continuation of it, such that our trust model for SCUs adds further details that strengthens the effectiveness of SCU management.

Trust in Social Computing Trust as a computational concept is first introduced by Marsh in [12], while Urbano

models Social Computational Trust in [19]. Crowdsourcing is the main paradigm that utilizes human capabilities in online task execution. Different context-based platforms such as those enabling pure social networking and those enabling networking in terms of expertise are an important part for provisioning human capabilities. Thus, there is a solid amount of work that investigates trust for human computation from the pure crowdsourcing perspective, modeling individual worker trust, as well as work that mix crowdsourcing with context-based networking, such as those in expert networks. Golbeck in [7], describes the TidalTrust algorithm for computing trust in online social networks, which is based on the breadth-first search. [16] present a framework to analyse trust in mixed service oriented networks, metrics and rules for inferring trust. They solve the issue of overloaded high-trusted workers with delegations. We believe that our approach, further strengthens trust models that consider delegation mechanisms because delegations are more efficient with our willingness confidence metric. Most trust models concerning social computing omit the aspect of people collaborating for executing complex tasks and their semi-automatic elastic management. Rather, most of the existing work focuses on initial collaborator/partner selection and more often analyze trust on the individual within social networks, which include for example the exchange of eCommerce goods in electronic marketplaces [21], user trust in mobile networks [11], or simple non-complex task executions such as those similar to HITs in Amazon Mechanical Turk. Some work have modeled trust in group collaborations, such as [3]. The difference of our work is that it concerns complex socio-technical systems, where collaborations are automatically managed with input from a human-in-the-loop.

Agent-based Trust Multidimensional trust inference for selecting a partner for collaboration is described in [10]. The mechanism of trust and reputation is based on multiple sources, such as from direct interactions, and trusted recommendations from direct and indirect communication that they name as witness reputation. [1] also considers indirect trust, where one agent is trusted to give recommendations about others within a specific context.

VIII. Conclusions and Future Work

We presented our investigation on the need for a combined SCU trust involving both social and automatically measured performance trust, and proposed a trust model to be used in SCU formations as well as in selecting already existing SCU structures for task executions. For this, we have defined and used performance metrics taking into consideration the freshness of performance data to develop trust metrics. An overall SCU trust metric is important for selecting an already existing SCU and avoiding the SCU formation step, as a speeding up of SCU invocations. Our example implementation of the model showed that when SCU adaptations are based on trust scores that include performance as well as social metrics, they are effective in keeping performance at more constant values and within a desired range. Our future work includes investigation on Service Level Agreements for elastic SCUs.

Acknowledgment

This work is supported by the EU FP7 FET SmartSociety project (n.600854, http://www.smart-society-project.eu/).

References

[1] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 6 - Volume 6*, HICSS '00, pages 6007–, Washington, DC, USA, 2000. IEEE Computer Society.

[2] M. Allahbakhsh, A. Ignjatovic, B. Benatallah, S.-M.-R. Beheshti, E. Bertino, and N. Foo. Reputation management in crowdsourcing systems. In *CollaborateCom*, pages 664–671. IEEE, 2012.

[3] X. Cheng, A. Azadegan, and G. L. Kolfschoten. An evaluation of trust development in group collaborations: A longitudinal case study. volume 46 of *Hawaii International Conference on System Science*, pages 78–85. IEEE computer society press, 2013.

[4] S. Dustdar and K. Bhattacharya. The social compute unit. *IEEE Internet Computing*, 15:64–69, May 2011.

[5] S. Dustdar and H. L. Truong. Virtualizing software and humans for elastic processes in multiple clouds- a service management perspective. *IJNGC*, 3(2), 2012.

[6] P. Fernandez, H. L. Truong, S. Dustdar, and A. R. Cortés. Programming elasticity and commitment in dynamic processes. *IEEE Internet Computing*, 19(2):68–74, 2015.

[7] J. Golbeck. Computing with trust: Definition, properties, and algorithms. In *Second International Conference on Security and Privacy in Communication Networks and the Workshops, SecureComm 2006, Baltimore, MD, Aug. 28 2006 - September 1, 2006*, pages 1–7, 2006.

[8] J. A. Golbeck. *Computing and Applying Trust in Web-based Social Networks*. PhD thesis, College Park, MD, USA, 2005. AAI3178583.

[9] Y.-T. C. Hung, A. R. Dennis, and L. Robert. Trust in virtual teams: Towards an integrative model of trust formation. In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 1 - Volume 1*, HICSS '04, pages 10042.1–, Washington, DC, USA, 2004. IEEE Computer Society.

[10] S. N. Keung and N. Griffiths. Trust in agent societies. chapter Towards Improved Partner Selection Using Recommendations and Trust, pages 43–64. Springer-Verlag, Berlin, Heidelberg, 2008.

[11] J. Li, Z. Zhang, and W. Zhang. Mobitrust: Trust management system in mobile social computing. In *CIT*, pages 954–959. IEEE Computer Society, 2010.

[12] S. P. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, 1994.

[13] D. Miornadi and L. Maggi. Programming social collective intelligence. In *Technology and Society Magazine, IEEE*, volume 33, pages 55–61. IEEE, 2014.

[14] A. J. Quinn and B. B. Bederson. Human computation: A survey and taxonomy of a growing field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 1403–1412, New York, NY, USA, 2011. ACM.

[15] M. Riveni, H.-L. Truong, and S. Dustdar. On the elasticity of social compute units. In M. Jarke, J. Mylopoulos, C. Quix, C. Rolland, Y. Manolopoulos, H. Mouratidis, and J. Horkoff, editors, *Advanced Information Systems Engineering*, volume 8484 of *Lecture Notes in Computer Science*, pages 364–378. Springer International Publishing, 2014.

[16] F. Skopik, D. Schall, and S. Dustdar. Trustworthy interaction balancing in mixed service-oriented systems. In S. Y. Shin, S. Ossowski, M. Schumacher, M. J. Palakal, and C.-C. Hung, editors, *SAC*, pages 799–806. ACM, 2010.

[17] SmartSociety. Hybrid and diversity-aware collective adaptive systems: When people meet machines to build a smarter society. http://www.smart-society-project.eu/. FP7 FET,EU Funded Project,Accessed: 2014-10-20.

[18] H.-L. Truong and S. Dustdar. Context-aware programming for hybrid and diversity-aware collective adaptive systems. In F. Fournier and J. Mendling, editors, *Business Process Management Workshops*, volume 202 of *Lecture Notes in Business Information Processing*, pages 145–157. Springer International Publishing, 2015.

[19] J. Urbano. *A Situation-aware and Social Computational Trust Model*. PhD thesis, 2013.

[20] M. M. Wasko and S. Faraj. Why should i share? examining social capital and knowledge contribution in electronic networks of practice. *MIS Q.*, 29(1):35–57, Mar. 2005.

[21] G. Zacharia, A. Moukas, and P. Maes. Collaborative reputation mechanisms for electronic marketplaces. pages 371–388, 2000.

[22] J. Zhan and X. Fang. Trust maximization in social networks. In J. Salerno, S. Yang, D. Nau, and S.-K. Chai, editors, *Social Computing, Behavioral-Cultural Modeling and Prediction*, volume 6589 of *Lecture Notes in Computer Science*, pages 205–211. Springer Berlin Heidelberg, 2011.