Calculating and Aggregating Direct Trust and Reputation in Organic Computing Systems

Rolf Kiefhaber
Institute of Computer Science
Augsburg University, Germany
E-Mail: kiefhaber@informatik.uni-augsburg.de

I. Introduction

The Organic Computing Initiative [1] identified the growing complexity of modern system as one of the big current challenges. These systems consist of a rising number of interacting parts, whose interactions increase in complexity as well. The Organic Computing Initiative aims to control these complexities by introducing so called self-x properties. The basic idea is to self-configure, self-optimize, self-heal and self-protect these systems. These properties are achieved by constantly observing the system and initiating autonomous reconfigurations when necessary (observer/controller paradigm [2]). By enabling autonomous reconfigurations Organic Computing Systems are able to react on disturbances without the immediate intervention of a user.

So far, Organic Computing Systems assume the benevolence of every involved interaction partner to obtain a more robust system utilizing these self-x properties. In open heterogeneous systems, like in cloud or grid computing, this benevolence assumption can no longer hold. In such systems, participants can enter and leave the systems at will. In addition, not every participant is interested in an altruistic cooperation to further the system goal. Some participants might try to exploit the systems or even try to attack and disrupt it.

By incorporating trust, the behavior of the participants can be monitored and identified. By utilizing this information the self-x properties of Organic Computing Systems are able to consider the behavior of its participants and are therefore able to maintain a more robust configuration in the face of unreliable components. This enables a reliable system out of unreliable components.

When speaking of trust, several definitions can be found in current literature. This dissertation is part of the research unit OC-Trust of the German Research Foundation (DFG). We published our definition of trust in [3]. We see trust as a multi-faceted multi-contextual subject and therefore defined the following facets:

- **Functional correctness:** The quality of a system to adhere to its functional specification under the condition that no unexpected disturbances occur in the system's environment.
- **Safety:** The quality of a system to be free of the possibility to enter a state or to create an output that may impose harm to its users, the system itself or parts of it, or to its environment.
- **Security:** The absence of possibilities to defect the system in ways that disclose private information, change or delete data without authorization, or to unlawfully assume the authority to act on behalf of others in the system.
- **Reliability:** The quality of a system to remain available even under disturbances or partial failures for a specified period of time, measured quantitatively by means of guaranteed availability, mean-time between failures, or stochastically defined performance guarantees.
- **Credibility:** The belief in the ability and willingness of a cooperation partner to participate in an interaction in a desirable manner. Also, the ability of a system to communicate with a user consistently and transparently.
- **Usability:** The quality of a system to provide an interface to the user that can be used efficiently, effectively and satisfactorily that in particular incorporates consideration of user control, transparency and privacy.

I focus on calculating reliability of nodes in a distributed network. When calculating trust, two categories have to be considered: Direct Trust and reputation.

- **Direct Trust** describes the trust one builds with an interaction partner based on its own experiences.
- **Reputation** stands for recommendations of third parties, i.e., the trust others had with my interaction partner.

In my thesis I investigate and research trust metrics to calculate direct trust, reputation and an aggregated total trust value from these two parts. The metrics are based on the trust definition mentioned above with focus on the facet reliability. The nodes form a heterogeneous open system. Each node is able to host some kind of service that provides functionality to use within the system. Integrating self-x properties in such a system enables a robust distribution of the services. Utilizing trust in this system, focused on the reliability of the nodes, enables a more robust distribution of the services, because unreliable nodes as well as node failures can now be considered when distributing services. It is thereby possible to rank the services by their importance and assign the more important services to more reliable nodes. Important services are those, which are essential for the functionality of the overlaying application. E.g., Bernhard el al. [4] present a computing grid to calculate big, yet parallelizable computational problems in a Multi-Agent System (MAS), which incorporates trust to form trusted communities (TCs). The managers, that administrate these TCs, are an example for an important service, since the

failure of a manager cripples the entire TC.

In my work I observe the behavior of nodes within a middleware. I assume that every node is equally able to implement the self-x properties. Therefore specific nodes for the self-x properties are not required, since each node implements the trust metrics and the algorithms for the self-x properties. In addition, I investigate systems, where the reliability of the nodes can be different. If all nodes would be reliable, nothing bad could actually happen or would be quite unlikely, e.g, a node failure, therefore no trust would be needed. The middleware system investigated in this work, the so called *Trust Enabling Middleware (TEM)* [5], is supposed to handle unreliable components and can be applied to any kind of distributed system, i.e., Multi Agent Systems (MAS). The TEM implements the algorithms developed in this dissertation and provides interfaces to allow all applications running on the TEM to use these algorithms.

II. Metrics

To calculate the trust values required for the self-x properties, four different parts have to be considered:

1) **Direct Trust**: First, the reliability of the nodes has to be observed and calculated. This is the basis for the other trust metrics and the decisions of the self-x properties.
2) **Reputation:** If the personal experiences with other nodes are not adequate enough to form a consistent decision, the experiences of other nodes have to be obtained. Therefore a reputation mechanism has to be defined.
3) **Confidence:** Before both values, direct trust and reputation, can be aggregated to a total trust value, the reliability of one's own trust value has to be determined, the so called *confidence*. If a node does have a direct trust value but is not confident about its accuracy, it needs to include reputation data as well.
4) **Aggregation:** When all the aforementioned values are obtained, a total trust value based on the direct trust and reputation values can be calculated using confidence to weight both parts against each other. This value can then be used to improve the self-x properties.

While direct trust developed in this dissertation is focused on obtaining the reliability of nodes, the reputation, confidence and aggregation are applicable to all kinds of direct trust values of any facet. The metrics are generic enough to achieve this goal.

A. Direct Trust

The basis for the trust value is the direct trust, the trust based on the direct experiences of a node. For an improvement of the self-x properties an evaluation of the reliability of a node is required. Such an estimation has to be done without knowledge about the functionality of the distributed services, since this estimation is done on middleware level. Nevertheless the reliability of a node can be measured by observing the message flow to other nodes. If messages are lost, either the node or the connection to it is unstable or has failed. In this case the reliability of the node is rated down and it is no longer appropriate for important services, because messages targeted to such a service might be lost as well. Additionally the loss of messages might refer to an error in the node itself and its imminent failure. In this case an important service running on it would fail as well.

B. Reputation

If no direct experiences could be obtained, and therefore no direct trust value calculated, or if the direct trust value is not yet sufficient enough, other nodes that already had experiences with a node are asked about their opinion. The total amount of all opinions of other nodes forms the reputation value for the potential interaction partner. A node n_1 that has direct experiences with another node n_2 is called a *neighbor* of n_2. An important aspect of the reputation metric is to separate the direct trust value of a node in any context from its ability to provide appropriate reputation data. Marmól and Pérez [6] demonstrated attack scenarios, which are only possible, if these values are not separated. In general, the reliability of a node says nothing about the accuracy of its recommendations. The recommendation of a neighbor can be weighted for the total reputation value (consisting of the recommendations of all neighbors about a node) regarding its previous recommendations. When the information of a neighbor has proven to be false, e.g., the node tries to disrupt the system by providing incorrect information, its future recommendations are rated down for the total reputation value. Using this method, lying nodes can be identified and their incorrect recommendations discarded. This also means, that nodes can redeem themselves by providing correct information in the future.

A neighbor which recommendations differ from a node's own experiences is not necessarily malicious. Its experiences might be different from the nodes, e.g., the connection between it and its neighbor is unstable. Its recommendation is nonetheless of no use for the node, since the direct experiences of the node take precedence to decide whether to actually interact with the interaction partner. Golbeck [7] demonstrated in her evaluation scenario, which consisted of a movie rating platform combined with trust relations between the raters, that getting recommendations from others, that have similar experiences, or in this case taste in movies, is superior than using the opinion of the masses.

The thresholds, when an recommendation is similar enough to one's own experience, are adjustable as well as the amount of maximal adjustment of the weight, be it positive or negative, is adjustable. Therefore the metric can be adjusted to any kind of application scenario.

C. Confidence

An estimation about the accuracy of one's own trust value is required, before both values can be aggregated. This estimation is done by calculating the *confidence* of the direct trust value. With a high confidence, the direct trust will be rated higher than reputation in the total trust value and vice versa. The confidence rates three different aspects of the experiences that were used to calculate the trust value:

- **Number**: Very few experiences are not suitable to express the actual behavior of an interaction partner, especially when its behavior contains some variance.
- **Age**: Older experiences might be outdated when interaction partner are able to change their behavior. Such outdated data might reflect its past behavior but not the current one.
- **Variance**: Since trust values are typically calculated by a mean or weighted mean metric a high variance of the experiences is not reflected in such a mean value. Therefore it is important to consider the variance as well.

The total confidence is a weighted mean of these three aspects, whereas the weights are able to be adjusted by the application to adhere to scenarios, where one or more of these aspects are less important than the others. Additionally the number and weight confidence are adjustable as well. For the number confidence, the threshold, when enough experiences are gathered, is adjustable. For the age confidence, the thresholds, when an experience is completely outdated or completely up to date is adjustable as well.

D. Aggregation

With the confidence a weight for the aggregation using a weighted mean of direct trust and reputation can be calculated. A high confidence results in a high weight for the direct trust value and vice versa. Here the question is to find a good formula to calculate the weight from the confidence. Figure 1 illustrates the function I want to use to archive this goal. When looking at human trust decisions most of it is done intuitively without any form of quantification. In my thesis I plan to quantify the point, when to switch from reputation to direct trust, or more precisely, how to calculate the weight between direct trust and reputation. In my formula this means to find good values for the two thresholds τ_{cl} and τ_{ch}.

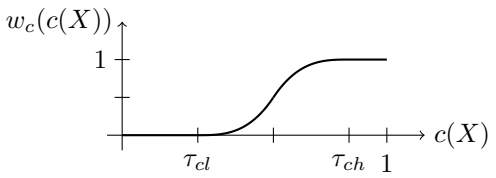

Fig. 1. Function to calculate the weight $w_c(c(X))$ for the total trust value based on the confidence $c(X)$ of all experiences X

III. Related Work

Trust is an actively worked on research fields with a plethora of different metrics. Many of them include users for direct trust. Below are some of the more prominent trust frameworks presented.

SPORAS [8] is another reputation metric. Its focus is to prevent entities to leave and rejoin the network to reset possible bad reputation values. Compared to my reputation metric, SPORAS does not assign different values for the reputation value provided by another interaction partner and the trustworthiness of that interaction partner to give accurate reputation data. The trustworthiness is calculated from its reputation value. I differentiate between these values by defining separate weights; Mármol and Pérez [6] have shown the importance to do this.

FIRE [9] is a trust framework combining direct trust and reputation (called *witness reputation* in FIRE). In addition, it adds the trust parts of *certified trust* and *role-based trust*. Certified trust describes past experiences others had with an agent, who can present it as reference of his past interactions. Role-based trust stands for generic behavior of agents within a role and the underlying rules are handcrafted by users. The four parts are then aggregated with a weighted mean, whereas the weights are adjusted by a user depending on the current system. In comparison, my work does not require user handcrafted parts like the role-based trust of FIRE and is therefore able to run in a fully automated environment.

ReGreT [10][11] is a trust framework providing similar metrics for direct trust, reputation, and aggregation to my metrics. Some differences to my work exist. The age of experiences is part of the direct trust calculation whereas I have the age, number and variance as confidence (called the *reliability of the trust value* in ReGreT). Additionally, my metrics for the confidence metrics are parametrized. Similarly, my reputation metric can be parametrized to define the threshold, when one's own experiences are close enough to the reputation data given by a neighbor (called a *witness* in ReGreT). Also I do not use the confidence directly for the aggregation but a parameterizable function to calculate the weight for using direct trust instead of reputation, beside using a non linear function to aggregate direct trust and reputation. One of the major differences though lies in the evaluation. While ReGreT works in a scenario with fixed agent behaviors I investigate systems with varying behavior, where a very trustworthy node can change to the direct opposite. Several such changes per scenario are considered by me.

I also investigate the impact of the parameters and identify appropriate parameter configurations by utilizing automatic design space exploration.

IV. Evaluation

To evaluate the different metrics, especially the aggregation, a scenario with a set of nodes is defined, where the nodes each have a mean reliability with a specific variance. These two values, mean reliability r and variance v, are generated randomly for a scenario, but within certain bounds, e.g., more reliable nodes have $r \in [0.8, 0.9]$ and highly unreliable nodes have $r \in [0.2, 0.3]$. This behavior is achieved by utilizing a beta distribution[1]. The result of each interaction, and therefore the rating of the experience, is taken from the beta distribution. Jøsang and Elouedi [12] presented subjective logic, which enriches binary logic with uncertainty and adds a complete algebra on it. They showed that subjective logic expressions can be bidirectionally translated to a beta distribution. Since trust is used to handle uncertainties, using a beta distribution for node behaviors is suitable. Additionally a beta distribution includes several other distributions, e.g., $\alpha = 1, \beta = 1$

[1] https://en.wikipedia.org/wiki/Beta_distribution

for mean distribution, so several possible behaviors can be modeled.

To investigate the effects of the different metrics, there is an amount of nodes to interact with n and some other nodes (the evaluation nodes) that interact with these nodes n_e. The simulation is divided into time steps, where each evaluation node is performing an interaction with one of the normal nodes. They consider their own experiences, as well as the information from the other evaluation nodes, to decide which node to interact with. Some nodes also change their behavior, i.e., changing the configuration of their beta distribution, several times in the evaluation. The evaluation nodes thereby try to obtain the best result from each interaction, which is called *benefit*. The benefit represents the rating of a single experience. It can be a number between 0 (worst possible result of the interaction) to 1 (best possible result of the interaction. After several time steps the total cumulative benefit is taken as fitness function to rate the effectiveness of each metric variant, since the goal of a node is to choose the participants that provide the best benefit and therefore highly profitable interactions. These variants include using only direct trust (DT), using direct trust and confidence (DTC) as well as combining all parts, that is direct trust, confidence and reputation (DTCR).

Since nearly all parts of the metrics can be parametrized, except direct trust, an *automated design space exploration (ADSE)* is applied to find suitable parameters for each metric as well as investigating the effects of poorly chosen parameters. An ADSE employs heuristic algorithms, like genetic or particle swarm optimization algorithms, to find good enough results in a parameter space that is too big to traverse completely. Thereby several scenarios of agent behavior should be investigated, including more frequent behavior changes or completely random behavior. An important aspect is also the selection metric used. To balance the exploration versus exploitation problem (when should unknown interaction partners be explored versus when to use already known interaction partners) a selection metric based on the roulette-wheel selection metric is applied.

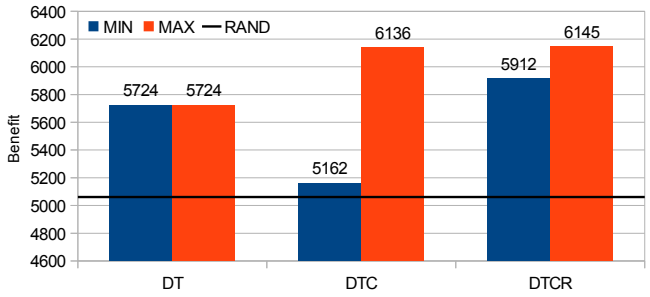

Fig. 2. Results when comparing the effectiveness of the different trust metrics with $n = 100$ and $n_e = 10$

Figure 2 shows some results of an evaluation when using only direct trust (DT), direct trust and confidence (DTC) as well as direct trust, confidence and reputation (DTCR) compared to random (RAND) to choose the next interaction partner. As was described before the goal was to maximize the total cumulative benefit over all interactions, in case of this simulation 8000. The metrics were parametrized using ADSE to find the worst possible solution (left column / MIN) and best possible solution (right column / MAX) for these parameters. It can be seen that using trust is better than random in all cases. Adding confidence to direct trust can increase the total benefit significantly but the benefit can get worse than by using direct trust alone, if unfitting parameters are defined. Adding reputation balances bad parameter choices while maintaining a high maximum result.

ACKNOWLEDGMENT

This research is sponsored by the research unit OCTrust (FOR 1085) of the German Research Foundation (DFG).

REFERENCES

[1] C. Müller-Schloer, "Organic computing: on the feasibility of controlled emergence," in *Proceedings of the 2nd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS 2004, Stockholm, Sweden, September 8-10, 2004*, A. Orailoglu, P. H. Chou, P. Eles, and A. Jantsch, Eds. ACM, 2004, pp. 2–5.

[2] U. Richter, M. Mnif, J. Branke, C. Müller-Schloer, and H. Schmeck, "Towards a generic observer/controller architecture for Organic Computing," in *Informatik 2006 - Informatik für Menschen, Band 1, Beiträge der 36. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 2.-6. Oktober 2006 in Dresden*, ser. LNI, vol. 93. GI, 2006, pp. 112–119.

[3] J.-P. Steghöfer, R. Kiefhaber, K. Leichtenstern, Y. Bernard, L. Klejnowski, W. Reif, T. Ungerer, E. André, J. Hähner, and C. Müller-Schloer, "Trustworthy organic computing systems: Challenges and perspectives," in *Autonomic and Trusted Computing*, ser. Lecture Notes in Computer Science, B. Xie, J. Branke, S. Sadjadi, D. Zhang, and X. Zhou, Eds. Springer Berlin / Heidelberg, 2010, vol. 6407, pp. 62–76.

[4] Y. Bernard, L. Klejnowski, J. Hähner, and C. Müller-Schloer, "Towards Trust in Desktop Grid Systems," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), 2010*, 2010, pp. 637–642.

[5] G. Anders, F. Siefert, N. Msadek, R. Kiefhaber, O. Kosak, W. Reif, and T. Ungerer, "TEMAS – A Trust-Enabling Multi-Agent System for Open Environments," Universität Augsburg, Tech. Rep. 2013-04, April 2013. [Online]. Available: http://opus.bibliothek.uni-augsburg.de/opus4/frontdoor/index/index/docId/2311

[6] F. G. Mármol and G. M. Pérez, "Security threats scenarios in trust and reputation models for distributed systems," *Computers & Security*, vol. 28, no. 7, pp. 545–556, 2009.

[7] J. A. Golbeck, "Computing and applying trust in web-based social networks," Ph.D. dissertation, University of Maryland, College Park, MD, USA, 2005.

[8] G. Zacharia, "Trust management through reputation mechanisms," *Applied Artificial Intelligence*, vol. 14, pp. 881–907, 2000.

[9] T. Huynh, N. R. Jennings, and N. Shadbolt, "An integrated trust and reputation model for open multi-agent systems," *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 2, pp. 119–154, 2006.

[10] J. Sabater and C. Sierra, "Social ReGreT, a reputation model based on social relations," *SIGecom Exch.*, vol. 3, no. 1, pp. 44–56, Dec. 2001. [Online]. Available: http://doi.acm.org/10.1145/844331.844337

[11] J. Sabater Mir, "Trust and reputation for agent societies," Ph.D. dissertation, Universitat Autònoma de Barcelona, 2002.

[12] A. Jøsang and Z. Elouedi, "Redefining material implication with subjective logic," in *Proceedings of the 14th International Conference on Information Fusion (FUSION 2011)*, 2011, pp. 1–6.