Chapter 1

Trust as an Interaction Mechanism for Self-Organising Systems

Giovanna Di Marzo Serugendo
Centre Universitaire d'Informatique
University of Geneva, Switzerland
Giovanna.Dimarzo@cui.unige.ch

Software applications are more and more decentralised, made of autonomous, sometimes roaming, entities or agents. They run in dynamic environments, where they interact with entities that are not known at design time. Applications enter into communication as human people, engage into discovery, negotiation, and transactions processes. They have to take decisions with local and incomplete knowledge about the capability and the trustworthiness of entities with which they interact. This paper proposes an engineering method for designing self-organising applications, based on the human notion of trust and grounded on an exchange of specification about entities capabilities.

1.1 Introduction

Emerging computing infrastructures are heterogeneous, ubiquitous and mobile. They are built on wireless network infrastructures that make it possible for devices to spontaneously interact. The environment available to a device is thus constantly changing. By their heterogeneity, scale, and dynamism, these systems gain to be designed so that they organise themselves autonomously. Interaction among devices that do not know each other can occur through some exchange of information specifying their respective capabilities. However, in a dynamic and unsecure environment, this is not sufficient. On the one hand, a malicious entity can exhibit desirable characteristics, while it is not able to realise them. On the other hand, even if in good faith, a printer can fail because it lacks toner or paper. The approach proposed in this paper is to combine functional information that entities carry about themselves (specification), with trust and recommendation information exchanged among entities about other entities. The specification is useful for entities to discover each other capabilities and functionality. Trust enables adaptation of running entities to the dynamic modifications of their environment. For instance, in the case of users and printers, a printer exhibits its characteristics, such as postscript, double-sided, black and white, and users exchange trust information about printers based on their observations and experiences realised with the printers, such as frequent paper jams, or low toner. Recommendations circulate among users about printers. Trust in a well functioning printer raises, while trust in an always broken down printer decreases.

This paper shows first that systems built on the human notion of trust fulfill the necessary requirements for self-organisation, established by Nobel Prize Prigogine. Second, it proposes an interaction mechanism based on a tag-based model where entities are equipped with a marking, carrying a specification of the functional as well as non-functional capabilities they offer to the community; and on a trust-based model, where entities build trust about other entities on the basis of experienced interactions or received recommendations. The paper demonstrates as well the approach through a small example involving users and printers.

1.2 Human Notion of Trust

Uncertainty and partial knowledge are a key characteristic of the natural world. Despite this uncertainty human beings make choices, take decisions, learn by experience, and adapt their behaviour. Most decisions implicitly rely on the trust that human beings have on their peers, their legal institutions, or business companies. A common example is provided by the trust put in banking establishments, acting as largely trusted third parties for credit card based interactions.

1.2.1 Trust-based engineered systems

Similarly to human beings, software entities taking part in decentralised and distributed systems, are autonomous, possibly roaming, and need to take decisions with local and incomplete knowledge. They are embedded in highly dynamic environments where peer entities appear and disappear permanently, and where information dynamically changes and is not permanently valid. Interactions with peers can occur only locally, there is only partial knowledge about the entities and about the environment. A trust-based schema helps entities evaluating the good faith or the correct functioning of a partner.

Therefore, systems that we consider are composed of a set of entities that interact with each other. These entities are autonomous components able to take decisions and initiatives, and are meaningful to trust or distrust. In the trust and security domains, such entities are called *principals*. Principals are for instance portable digital assistants (PDAs) acting on behalf of a human being, or personal computers, printers, mobile phones, etc. They interact by asking and satisfying services to each other.

In a system based on the human notion of trust [1], principals maintain local *trust values* about other principals. A principal that receives a request for collaboration from another principal, decides or not to actually interact with that principal on the basis of the current trust value it has on that principal for that particular action, and on the risk it may imply of performing it. If the trust value is too low, or the associated risk too high, a principal may reject the request. A PDA requiring an access to a pool of printers, may see its access denied if it is not sufficiently trusted by the printers. For instance, it is known that this PDA sends corrupted files to the printers.

After each interaction, participants update the trust value they have in the partner, based on the evaluated outcome (good or bad) of the interaction. A successful interaction will raise the trust value the principal had in its partner, while an unsuccessful interaction will lower that trust value. Outcomes of interactions are called *direct observations*. After interacting with a printer, a PDA observes the result of the printing. If it is as expected, for instance double-sided, and the document is completely printed, the PDA will adjust the trust value on that particular printer accordingly.

A principal may also ask or receive *recommendations* (in the form of trust values) about other principals. These recommendations are evaluated (they depend on the trust in the recommender), and serve as *indirect observations* for updating current trust values. As for direct observations, recommendations may either raise or lower the current trust value. We call *evidence* both direct and indirect observations. Some PDAs may experience frequent paper jams, on a given printer. They will update (in this case lower) their trust value in that printer, and advertise the others, by sending them their new trust value. The PDA that receives this recommendation will take it into account, and decide if it uses that printer or not [8].

Thus, trust *evolves* with time as a result of evidence, and allows to adapt the behaviour of principals consequently.

1.2.2 Trust as a self-organising mechanism

Nobel prize Ilya Prigogine and his colleagues have identified four necessary requirements for systems exhibiting a self-organising behaviour [4].

- Mutual Causality.
  "At least two components of the system have a circular relationship, each influencing the other" [2].

- Autocatalysis.
  "At least one of the components is causally influenced by another component, resulting in its own increase" [2].

- Far-from equilibrium condition.
  "The system imports a large amount of energy from outside the system, uses the energy to help renew its own structures (autopoeisis), and dissipates rather than accumulates, the accruing disorder (entropy) back into the environment" [2].

- Morphogenetic changes.
  "At least one of the components of the system must be open to external random variations from outside the system. A system exhibits morphogenetic change when the components of the system are themselves changed" [2].

We will now show for each point how the systems built on the human notion of trust address these four requirements.

- Mutual Causality.
  Principals exchanging recommendations about other principals influence the behaviour of each other. Principal A receiving a recommendation from recommender B about subject C, will update its trust value in C depending on the recommendation received and its own trust value in B as a recommender. Similarly, A sends recommendations to B about C or other principals.

  Additionally, interactions between two principals leads to direct observations from each of them, which cause an update of their respective trust value in the other.

- Autocatalysis.
  Principals exchanging references about other principals have an autocatalytic effect on the system, in the sense that a positive recommendation, received from one principal about another principal will reinforce the trust that the receivers has in that principal, while a negative recommendation will contribute to degrade that trust value. An increased trust value will increase the number of interactions with that principal, while a decreased trust value will lower the number of interactions. It is similar for direct observation leading to a positive or negative evaluation of the outcome of an interaction, causing a corresponding update of the trust value.

- Far-from equilibrium condition.
  Systems we consider are part of a highly changing environment. They need power supply, network links, memory, etc. Principals permanently join and leave the system (inserted, removed by the environment, or no longer available because of lack of resources). A trust-based system integrates new principals (builds trust values), or updates information and trust about malfunctioning or non-responding principals (autopoeisis).

Malfunctioning, non-responding or malicious principals see their accesses or interactions denied from other principals. In that sense, they get out of the set of interacting principals (maybe temporarily) and are then part of the environment, since they still consume some resources (entropy is pushed back into environment).

- Morphogenetic changes.
  Systems we consider are permanently faced with random conditions that affect both the environment, such as broken network links, joining and leaving entities, and the components themselves, e.g., low power or memory, paper jams, or low toner, or software evolution.

1.3 Tag- and Trust-based Model

Human beings exchange different kinds of *semantical* information for different types of purposes: to understand each other, to share knowledge about someone or something else, to take decisions, to learn more, etc. Despite people share the same understanding regarding information, this information remain local, incomplete and uncertain, leading people to rely on trust to actually take decisions. A traveler, booking and paying a plane ticket, implicitly trusts both the travel agency and the airline company that the flight exists and is correctly scheduled. Our model considers the two above aspects of human behaviour: (a) communication through semantical information; and (b) ability to take decisions despite uncertainty based on the notion of trust and risk evaluation.

The model defines a homogeneous framework which serves for expressing and checking semantical information of different kinds: functional behaviour, non-functional behaviour, observations, and recommendations. Principals carry a tag, a kind of marking, which contains this information. Before interacting principals exchange their respective tags. On the basis of the information contained in the received tag, on the current trust value, and past observations, principals decide to interact, to grant or deny access, or to ask for more information.

The semantical information is expressed using a higher-order logical language, it allows the definition of a theory, acting as a formal specification, comprised of: vocabulary, relations, axioms and theorems conveying the semantical part of the specification. This is useful for checking proofs at run-time. Proved theorems then assess semantical meaning. This allows also interaction with another principal if it can bring the proof (and that proof can be checked) that the way it intends to work corresponds to what is expected.

We describe here how interactions and trust-management are realised in this model:

- *Request for collaboration and exchange of tags.* A principal A receives a request for collaboration from another principal B. A and B exchange their respective capability tags under the form of a specification expressed in the logical language. They learn each other about their respective provided services.

- *Decision to interact.* Based on the received tag, A and B respectively evaluate if the services provided by the other fulfill its needs (proofs of theorems expected to be satisfied by the partner).

  The decision then depends on the evaluation of the tags, past direct observations of interactions with B (if any), previously received recommendations about B from other principals, current trust value A has about B, and the risk incurred by the interaction.

- *Trust Update.* If A decides to interact with B, it will observe the outcome of the interaction, evaluates it (positive or negative), and updates the trust value accordingly.

  Besides collaboration requests, principal A may receive a recommendation from B under the form of specification precising the degree of trust the recommender has on a subject C. Recommendations are evaluated with respect to trust in the recommender, and make the trust A has in the subject C evolve (increase or decrease).

The trust-based management part of this model has been fully implemented in the context of the SECURE project [1]. Preliminary work addressing interaction based on functional specification relying on a common ontology have been realised [7]. The integration of the trust-based model into a tag-based framework using higher-order logic is currently under work.

1.4 Example

The example presented here is commonly used to demonstrate interoperability of autonomous components. We consider a system composed of a group of computers and a group of printers. Those groups are not predefined, i.e., printers and computers can join or leave the system at any time. Before interacting with each other computers and printers exchange their respective functional as well as non-functional capabilities, e.g. a printer claims that it is a postscript double-sided printer, and a computer asks to print a PDF file. After having interacted with a printer, the computer stores the observation related to its experience with the printer (works as expected, only one side, no impression at all, etc.). Depending on the outcome of the interaction, or if it has been requested to do so, the computer may want to share its knowledge with some of the other computers. It will then inform the others that the printer is not actually double-sided, but only single sided, or that the printer went out of toner, and is no longer available, or that one of the printers is faulty and has a random behaviour.

1.4.1 Scenario

In the group of printers, there is one printer currently available, called lw6 (laser-writer nb 6). It is postscript, double-sided printer; PDAs, located close to lw6, begin to use the printer, but it appears that it has frequent paper jams, and that it is not able to print some PDFs files. A new printer from another manufacturer (lw3), more recent, is installed at another floor. PDAs using lw6 receive recommendations from other PDAs using lw3 that it is new, and can print any PDFs, particularly those not printable by lw6. Those PDAs leave lw6 and start printing on lw3. Later, lw6 is upgraded, and replaced by a printer of the same kind of lw3. The PDAs, noticing the change either because their user asks to print on lw6 or because they receive according recommendations, immediately start printing again on lw6. However, through experience, it appears that lw6 is not as good as lw3. Actual printing is random, PDAs are never sure that their jobs will be printed. However, before taking a decision to print on lw6 or lw3, PDAs evaluate the risk to lose the time of their user. From one hand, there are 50% chances of success to print on lw6, and lw6 is close to the user. On the other hand, a printing on lw3 will be successful, but the user will have to leave his office, walk down the stairs, and come back (he will surely lose 5 mins, instead of 30 sec if it works on lw6). Finally, a printer hidden in the library is discovered by some user (and consequently by its PDA), it is an old printer, but works well without paper jam, and is physically placed closer than lw3 for the users of lw6. Finally, software running in lw6 is updated, and the printer starts working correctly[1]. Besides, a malicious PDA floods all printers with big size documents consuming toner and paper. Printers exchange recommendations about that PDA, and decide to deny it access to their services.

1.4.2 Self-Organisation

Let us now examine the four requirements for self-organisation.

Mutual Causality: PDAs influence each other with their recommendations on the printers. In the case of the malicious PDA, it causes the printers to run out of paper, but in turn its access is denied.

Autocatalysis: PDAs experiencing troubles with lw6 receive reinforcing recommendations from other PDAs regarding the behaviour of that printer. When lw3 is installed, PDAs massively stop using lw6.

Far-from equilibrium condition: New printers and PDAs join and leave the system regularly (autopoeisis). The faulty printer lw6 is left apart from the system, it is no longer used. The malicious PDA has no longer access to the printers (entropy).

Morphogenetic changes: Printer lw6 has been updated two times. The first time it has physically changed, the second time, the software only has been changed.

We also observe emergent patterns of behaviour: The system is then composed of used printers, authorised PDAs that adapt their behaviour to changing conditions. We can also mention that in this scenario: (1) reputation emerges from recommendations (based on the notion of trust). It is largely known that lw6 is not reliable; and (2) group formation emerges from interactions (based on

[1] This scenario has been inspired from actual behaviour of printers and human beings in our department.

tags and trust), such as groups of PDAs that start or stop using some printer, groups of printers that exclude PDAs.

The model proposed here follows the separation into individual capabilities and social organisation mentioned by Minsky [6] (p.32). The exchange of functional and non-functional capabilities in our model corresponds to the diffusion of knowledge about the capabilities of individual principals. The use of trust and the exchange of recommendations adds a social layer on top of the interaction mechanism. Typical applications that can benefit from this technology include wireless cellular network routing, ambient intelligence systems [3], autonomic computing systems [5], or access control systems.

1.5 Acknowledgments

This work is partly supported by the EU funded SECURE project (IST-2001-32486), and Swiss NSF grant 21-68026.02

Bibliography

[1] Cahill, V., and al., "Using trust for secure collaboration in uncertain environments", *IEEE Pervasive Computing Magazine, special issue Dealing with Uncertainty* **2**, 3 (2003), 52–61.

[2] Contractor, N. S., and D. R. Seibold, "Theoretical frameworks for the study of structuring processes in group decision support system - adaptive structuration theory and self-organizing systems theory", *Human Communication Research* **19**, 4 (1993), 528–563.

[3] Ducatel, K., M. Bogdanowicz, F. Scapolo, J. Leijten, and J.-C. Burgelman, "Scenarios for Ambient Intelligence in 2010", *Tech. Rep. no.*, Institute for Prospective Technological Studies, (2001).

[4] Glansdorff, P., and I. Prigogine, *Thermodynamic study of structure, stability and fluctuations*, Wiley (1971).

[5] Kephart, J. O., and D. M. Chess, "The Vision of Autonomic Computing", *Computer* **36**, 1 (January 2003), 41–50.

[6] Minsky, M., *La Société de l'Esprit*, InterEditions (1988).

[7] Oriol, M., and G. Di Marzo Serugendo, "A disconnected service architecture for unanticipated run-time evolution of code", *IEE Proceedings-Software, Special Issue on Unanticipated Software Evolution* (2004).

[8] Terzis, S., W. Wagealla, C. English, P. Nixon, and A. McGettrick, "Deliverable 2.1: Preliminary trust formation model", *Tech. Rep. no.*, SECURE Project Deliverable, (2004).