2009

Enabling Adaptation in Trust Computations

Luca Longo
Technological University Dublin, luca.longo@tudublin.ie

Pierpaolo Dondio
Technological University Dublin, pierpaolo.dondio@tudublin.ie

Bresciani Riccardo
Trinity College Dublin, Ireland

See next page for additional authors

Recommended Citation

Longo, L., Dondio, P., Bresciani, R., Butterfield, A. & Barrett, S. (2009). Enabling adaptation in trust computations. *Computationworld 2009*, November 15-20, Athens, Greece, p.701-706. ISBN: 978-0-7695-3862-4. doi:10.1109/ComputationWorld.2009.70

Enabling Adaptation in Trust Computations

Longo Luca*, Dondio Pierpaolo*, Bresciani Riccardo†, Butterfield Andrew†, Barrett Stephen*

*Department of Computer Science and Statistics
Distributed Systems Group, Trinity College Dublin
{llongo, dondiop, stephen.barrett}@cs.tcd.ie

†Department of Computer Science and Statistics
Foundations and Methods Group, Trinity College Dublin
{bresciar, Andrew.Butterfield}@cs.tcd.ie

***Abstract*—Digital systems have been rapidly evolving within highly dynamic and unstructured environments, where the lack of a central authority forces entities to interact with each other through collaboration and negotiation. Digital agents often use Trust models in order to compute the level of trustworthiness of the partner they want to collaborate with. Unfortunately, due to the evolution speed of open and collaborative environments, the trustworthiness of an agent varies over time, and as a result, Trust models must be continuously adapted to the changing context. In this work we address the problem by presenting a self-adaptive model for Trust computations. In particular, the proposed methodology seeks to continuously align the trust model in force with the changing context in Web 2.0 dynamic applications such as forums, blogs, p2p systems. The self-adaptation is reflected in the auto-organisation of the Trust function to obtain an accurate degree of agents' trustworthiness.**

***Keywords*-Computational Trust; Adaptation; Multi-agent Systems; Web 2.0; Non-monotonic Reasoning;**

I. Introduction

Computational models of trust have emerged in the last decade with the aim of exploiting the human notion of trust in open and collaborative environments, motivated by the critical problems of identity management [2], privacy [3], decentralisation of control [4] and quality of information [13]. A computational model of trust is a system able to quantify a level of trustworthiness for entities acting in a specific domain. That value that can be used in the context of decision support tools for selection of resources and partners. Formally, as suggested by the Computational Trust literature, a degree of trustworthiness is a prediction or assessment about the ability of a trustee entity to fulfil the expectations of a entity [9] [6]. Due to the not-static property of trust, that changes over time, trust models should consider adaptive strategies to produce accurate and context-aligned degree of agents' trustworthiness.

In this paper we present a methodology to test the validity of a Computational Trust model and adapt it to the changing external environment. Trust is an adaptive concept itself and a key element for enabling good adaptation. Trust is adaptive mainly for two reasons. Firstly, trustee entities might change their behaviour, producing pieces of evidence that require their level of trustworthiness to be adjusted. Secondly, the external environment might change, producing constraints that might affect the way entities judge other entities' behaviors (and therefore their Trust levels), or affect the threshold required for collaboration. For instance, we might trust a trader when the stock market is bullish, but we might decide to distrust him in a bearish market because our Trust threshold is now set higher. Not only is trust adaptive, but in turn it has been always seen as a key element for a successful adaptation. A level of Trust can be used as a filter to select more reliable resources, decreasing the complexity of the environment and increasing the quality of interactions. These features allow systems to enable more reliable adaptation and filter potential harmful or even malicious ones. We study how a trust model can adapt to a dynamic distribution of Trust values among a population, that, as described above, might change due to the introduction of internal or external constraints. The input of our study is a generic Computational Trust function over a set of parameters. The study presented in this paper aims to answer the question:

> *given a certain distribution of Trust values among a population and given our model of trust, is the trust model able to predict such a distribution? Is it possible to parameterise the trust model, in order to maximise the accuracy of the system's predictions?*

Note how the problem has two layers: first, we wonder whether our model is in general consistent with the sampling and second if we can adapt it to better fit the given population. Our methodology, described in section III, addresses these two layers. It provides a tool to verify the validity of our trust model and it offers a method to adapt a model's parameters. It is based on the comparison between the trust distrbution given by the sampling and some a priori assumptions, and a distribution of Trust values produced by our computational model for a given set of parameters. The Trust model used for this evaluation is called LTTM [5] and it produces a Trust value for a given entity, at a given time, as a linear combination of four temporal Trust factors. These Trust schemes take into consideration entities's interactions over time, by producing a measure of activity and frequency within the environment and modelling the concepts of persistence and longevity.

This paper is organised as follows. In section II related works in the field of Trust and Adaptation are presented while in section III it follows the description of our theoretical adaptive model. In section IV an evaluation over a large online web-community is addressed and in section V our comments and future works conclude this contribution.

II. Related work

This study in abstract aims to investigate adaptation mechanisms for the computation of Trust. Social scientists [11] [9] [4] define trust as a dynamic adaptive concept. The adaptive nature of trust is evident when trust is derived from past evidence or probability-based computation. For example, according to the direct-experience paradigm, the Trust value for an entity is based on the outcome of past interactions. When a new interaction occurs, the Trust value is updated proportionally to the outcome of the most recent interaction. A memory factor is usually present, giving the following shape to the system:

$$T_{val}^{new} = m \cdot T_{val}^{old} + (1 - m) \cdot f$$

where m represents the memory factor (i.e. how the last interaction affects the new Trust value) as a measure of feedback. In the extreme case, when $m = 1$ the Trust value is not dependent on past interactions, while when $m = 0$ the Trust value is totally dependent on the last feedback f and the agent does not keep any memory of the past. A value between 0 and 1 represents a system that reacts to new interactions slowly (m small) or fast (m high). Usually, the choice of m depends on factors related to the environment and how rapidly the agent should adapt: for high volatile environments, where agents are likely to change rapidly, m should be chosen to be small, while in a stable environment m should be kept high to take advantage of past experience. An example of such linear feedback system is presented in the p2p-based Trust model of Wang [12] that exhibits the key elements of an adaptive system: a feedback sensitive function and a closed loop where values are updated. Other examples in Trust may be taken from probability-based approaches. Here the expected behaviour of a trustee is mapped to a probability function when enough past evidence is collected, and the future behaviour is predicted by applying probability calculus. Adaptation is usually implemented using Bayesian inference, such as in [8]. In [13] the authors show how Computational Trust is a non-monotonic phenomenon that can change drastically and suddenly, and therefore required a mechanism to quickly adapt to the new situation. Recommendation and feedback systems provide an example of bad adaptation and a slow-to-react property. The usual global sum strategy used to aggregate feedback produces values that have been proven to be positively-biased and so highly insensitive to mutual dependent assertion in the domain. Finally, in the field of economics, Gorobets [7] defined an adaptable model of Trust for multi-agents systems. The adaptation is based on the realised profit agents gain by trusting other agents' advices.

III. The theoretical adaptive-model

The theoretical adaptive model presented here is intended to be a general schema: it can be extended or applied to different Trust schemes. Here we adopt the temporal factors presented by Longo et al. in their LTTM Computational Trust model [5] which considers temporal properties as pieces of Trust evidence to compute the trustworthiness of Wikipedia project articles and users. The Trust schemes adopted define real numbers in the interval $[0, 1]$:

- *Activity Factor* ($A(\gamma)$) The *activity factor* computes the activity percentage of the input agent γ to the total system activity at a τ time. It models the number of interactions concerning a given agent. This factor represents an interesting property that allows us to compare the activity of an agent is on average or not, compared to others agents involved in the modelled context.
- *Presence Factor* ($P(\gamma)$) The *presence factors* returns a measure of period length of the input agent γ at a τ time. It tries to model the human notion of experience over time: old agents are usually considered to be trusted.
- *Frequency Factor* ($F(\gamma)$) The *frequency factor* computes the frequency percentage of the input agent γ at τ time using the *awaited frequency constant* π which is a time interval indicating the expected frequency activity of each agent.
- *Regularity Factor* ($R(\gamma)$) The *regularity factor* models the concept of persistence and returns the regularity percentage of the input agent γ at τ time using the *awaited frequency constant* π. An agent is 100% regular if in each sub-interval of its life cycle, with dimension π, there exists at least one interaction with another agent, otherwise the agent is said to be irregular.

These factors are aggregated by a Trust function in order to assess the degree of trustworthiness of digital virtual identities. In the original LTTM model these Trust schemes are aggregated by using a ranking Trust function that computes the average of the rank position of each Trust property for each digital entity. In our methodology, during the formalisation of a Trust function a set of rules will be designed and their application will be reflected in the penalisation of certain behaviors and the awarding of other ones. For example, in the LTTM model, an entity with a high presence value may be considered as a trustworthy entity. A more accurate reasoning process will take into account more complex rules that analyse the possible contradictions or the supporting strength of group of more basic rules. For instance, the fact that an entity has high presence value, but low activity, does suggests that it is trustworthy. These facts interact negatively in designing trust scheme. We want therefore to be sure that such rules are consistent among themselves and do not contradict each other. This consistency test may not be straightforward if a large number of rules describe the environment modelled. For this reason a theorem prover can efficiently handle this issue.

In this example, we take into consideration the four LTTM's temporal factors and we compute a cross tabulation among each of them as in table I. The set of 4^2 rules describes all the possible combinations among the factors by assigning the related class of behaviours a boolean value (HIGH (H) or LOW (L)) for each factors. The threshold to decide whether an entity is HIGH or LOW may be the median, the average of the distribution for each Trust scheme or a more complex function that considers the standard deviation and further properties. Obviously it is possible to adopt a higher granularity with intermediate levels, but we want to keep the model simple to exemplify its capabilities. The literature in Computational Trust suggests the following five clusters:

- Very Trustworthy (VT);
- Trustworthy (T);
- Neutral (N);
- Untrustworthy (U);
- Very Untrustworthy (VU).

In natural language, each combination among the temporal factors has a particular meaning, extracted by reasoning on them. For instance, if an entity has a high degree of activity and presence within the system but low frequency and regularity, it may not mean it is a trustworthy entity. The fact that it has experience, with a high number of interactions in the environment, but also has a low frequency and does not interact regularly, results in an 'Untrustworthy' (U) judgement. Similarly, even if it shows a low degree of activity compared to others entities but it interacts regularly within the system with a high value of frequency, it may be considered as a 'Very Trustworthy' (VT) entity. The inference rules that emerge from the reasoning process used to model an environment, are of two types. The first type are Horn clauses, that simply associate a Computational

AP\FR	LL	LH	HH	HL
LL	VU	VU	U	VU
LH	VU	T	T	U
HH	N	VT	VT	T
HL	U	N	N	N

Table I
TEMPORAL BEHAVIORS AND RELATED CLASS OF TRUSTWORTHINESS. (A→ACTIVITY, P→PRESENCE, F→FREQUENCY, R→REGULARITY)

Trust class a certain behaviour (i.e. a certain combination of Computational Trust schemes). The second type states that rules must be coherent (i.e. it must not be possible to infer two different trust values for the same behaviour). The validation of inference rules can be performer by means of a first order logic theorem prover. This test is successful if the rules describe a partial function: it is not possible that two different rules exists such that a user behaviour can lead to two different Trust classes. A general rule describing a user's behaviour is a Horn clause:

$$T(A,P,F,R) \rightarrow TrustClass$$

The rules that impose coherence are of the following kind:

$$TrustClass_1 \wedge TrustClass_2 \rightarrow FALSE$$

If a theorem prover cannot find a contradiction for any user behaviour, *i.e.* to derive $FALSE$, we have proved the coherence of the rules that describe the requirements of the system.

During the formalisation of a Trust function, it is generally unrealistic to assume that all the Trust schemes support the final degree of entities' trustworthiness equally strongly. The Trust function need to be self-adaptive and each Trust scheme should have its own weight as in the follow:

$$T(\gamma) = W_a \cdot A(\gamma) + W_p \cdot P(\gamma) + W_f \cdot F(\gamma) + W_r \cdot R(\gamma)$$

where γ is a given agent, and W_x is the weight of the Trust scheme $X(\gamma)$. Unfortunately, it is often difficult to estimate a priori the strength of each Trust scheme due to the possible high dynamism and the unstructured property of the environment modelled. For this reason, our approach starts with a blind-aggregation of the Trust Factors, i.e. it assumes that all the Trust schemes contribute equally strongly to compose the final level of entity's trustworthiness.

During the computation of agents' Trust values, each Trust scheme produces a different distribution with unscaled non-normalised output. Consequently, all the distributions obtained by considering all the entities involved in the system, with their interactions, at a time τ, have to be normalised in order to effectively exhibit the same scale. A simple method is the popular Min-Max algorithm [1].

$$X_i^{'} = (max_t - min_t) \cdot \left[\frac{(X_i - min_v)}{(max_v - min_v)} \right] + min_t$$

where $X_i^{'}$ is the new scaled value, X_i is the value we want to normalise, max_t and min_t are respectively the target maximum value and the target minimum value, max_v and min_v are the maximum and minimum values in the whole distribution, for a given Trust Scheme.

After the scaling process, we obtain four scaled distribution, by considering the four LTTM's temporal schemes. The aim is to produce a final aggregated distribution where the results are ranked according to the sample chosen a priori to model the environment. To evaluate the effectiveness of the methodology, a comparison, for each entity, is performed. The entity's Trust Class obtained from the distribution produced by the Trust function $T(\gamma)$ is compared to the Trust Class inferred via the inferences rules as described in table I. The fraction of entities that have the same Trust Class represents a possible indicator of how good the system is. By taking into account the number of entities whose class have been correctly identified it is possible to derive the degree of reliability of the Trust function as a mean to correctly classify entities.

Due to the blind-aggregation adopted for aggregating the Trust schemes, the reliability level that would emerge by the class-matching process would likely be low. A self-adaptive algorithm completes the description of our proposed solution. The goal of this adaptive function is to adapt Trust schemes' weights in order to increase the reliability degree of the Trust function. It may be repeated until the class-matching process produces a percentage of class-matched entities greater than a confidence threshold decided upon, or it may take the configuration of Trust schemes' weights which produces the highest rate of matching. The methodology proposed in this study is summarised in figure 1 and in the following pseudo-code:

```
01. Conceptualisation and formalisation of Trust Schemes;
02. Formalisation of the possible Trust Classes;
03. do {
04.   Formalisation of the rules to aggregate the Trust
      schemes in 1 to produce Trust classes in 2;
05. }
06. until ('Test of  contradictions of rules in 4,
           by using a Theorem prover' == TRUE)
07. Computation of Trust Schemes, for each entity
    at a given time;
08. Extraction of Trust Classes, for each entity,
    by using the rules in 4;
09. Normalisation/scaling of Trust schemes' values in 7;
10. Definition of the Trust function by adopting equal
    weights for each Trust scheme;
11. Definition of the sampling distribution;
12. Computation of Trust values by using the Trust
    function in 10;
13. Extraction of entities' Trust Class
    according the distribution in 11;
14. For all entities, matching of Classes in 8 and 13;
15. if (matching % < Confidence Threshold) {
```

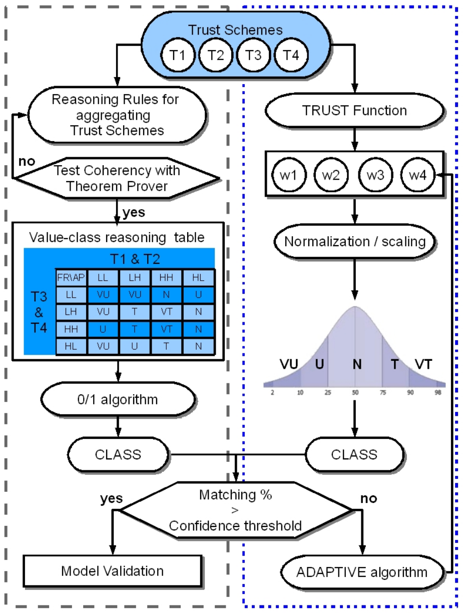

Figure 1. Trust Function's self-adaptive process

```
16.   do {
17.     self-adaptive algorithm to change the
        weights of the Trust function in 9 in order to
        formalise a new Trust function;
18.     Computation of new Trust values by using the
        redefined Trust function in 17;
19.     Extraction of entities' Trust class
        according to the distribution in 11; }
20.   until (matching % > Confidence Threshold)
21. }
22. Trust Function validation
```

A. Adaptive algorithm

The problem of finding the best values for each Trust schemes' weights can be seen as a global optimisation problem in a multi-dimensional space. The aim is to learn the combination of the weights that minimises the error in the Trust class assignments, so that the highest number of entities are in the same Trust class for both the computations (Inference rules distribution and Trust Function distribution).

Our search space is represented by the four temporal schemes. If we suppose that each temporal factor is bounded in $[0, i]$ our space is $[0, i]^4$. The adaptive algorithm is a simple heuristic that improves the basic exhaustive search by selecting at each interaction only promising regions of the search space. By choosing a resolution R in $[0, i]$ (where i is a multiplier of R) and a threshold T in $[0, 1]$, the algorithm scans the whole space sampling points by incrementing each dimension by $\frac{i}{R}$. The number of first iterations is therefore $(\frac{i}{R})^4$. For each point, corresponding to a specific Trust function, a metric is collected, such as the number of entities correctly matched (i.e. in the same Trust class). Other metrics are possible, for instance one may consider only the trustworthy matched entities. The heuristic discards all the points, i.e. a possible combination of weights, that are not in the top T-percentile positions according to the choosen metric. At the next interaction, the heuristic scans the region around each of the $T \cdot (\frac{i}{R})^4$ points left. The size of the region is set to $(\frac{i}{R})$, the previous interval, and the size of the increment is now $(\frac{i}{R})^2$. The process is iterated until the required resolution is reached. If i is the size of each dimension, R the resolution and T the percentile-threshold, n the number on interactions required, the final resolution will be $\frac{1}{R^n}$ and the number of interactions can be seen as $(\frac{i}{R})^4 + (n-1) \cdot T \cdot (\frac{i}{R})^4 \cdot (\frac{i}{R})^4$ compared to the $((\frac{i}{R})^n)^4$ all possible combinations.

For instance, if

- the size of each dimension i is $[0, 1]$;
- our required resolution is $R = 0.001$;

the exhaustive searching requires $(\frac{1}{0.001})^4$ = 10^{12} tests.

If

- a resolution $R = \frac{1}{10}$ and therefore $n = 3$ to have the adequate resolution $(\frac{1}{10^3})$;
- a threshold $T = 0.01$;

the number of interactions is: $10^4 + (3-1) \cdot 0.01 \cdot 10^4 \cdot 10^4$ i.e. around 10^6 interactions.

The resolution R may be changed at each interaction as the threshold T.

IV. Evaluation

The dataset used for the evaluation has been extracted from *Finanzaoline* the popular Italian finance forum [10]. conducted over more than $30,000$ users, almost $1,000,000$ threads and more than $11,000,000$ messages posted since 1999. We applied our heuristics approach to approximate the best choice of the four LTTM's Trust schemes in order to maximise the accuracy of our model. The *awaited frequency constant* was set to 1 day and the parameter chosen for our heuristics where the following:

- $[0, 1]$ for the parameters intervals;
- $R = \frac{1}{10}$ for the resolution at each interaction;
- $n = 3$ for the number of interactions;
- $T = 10^{-3}$ for the percentile threshold, meaning that only the first 10 points will be selected for the subsequent interaction.

We performed a sampling over the forum population, by classifying the trustwhortiness of each members based on an explicit pool promoted in collaboration with the forum administrators. We used a 5-tier classification and the percentile spread of the sampling population is described in table II. In order to apply our set of rules, as explained in the previous section, we defined the following threshold for each Trust factor: $T = average + standard\ deviation$.

Trust class	Percentile	No. of Users
VT	0.01	313
T	0.09	2817
N	0.2	6260
U	0.2	6260
VU	0.5	15653

Table II
Computed users - Trust Model with relative percentile

For the *activity factor*, whose distribution results were affected by a very high variance, we decided to set the threshold for this factor to $T = average$.

Our choices for the thresholds, coupled with to the set of rules identified, give a distribution of Trust classes depicted in table III.

Trust class	Percentile	No. of Users
VT	0.011	342
T	0.006	216
N	0.120	3714
U	0.040	1259
VU	0.823	25774

Table III
Computed users - rules, thresholds with relative percentile

In this initial experiment, the classes vary significantly in size with an exception for the class of very trustworthy entities (313 $vs.$ 342). The sampled population does not seem to behave as we should expect, apart from the very trustworthy entities. Another choice of the tresholds could be possible as long as it is based on reasonable assumptions, but the point is that the methodology has produced its first output, suggesting that the model does not predict correctly the sampled population. We keep our choice of threshold and we now evaluate the impact of the choice of parameters. Our evaluation considers what are the best choices for the parameters so that the highest number of entities are placed in the same Trust class for both the computations.

We performed experiments with the following metrics:

- M is the percentage of overall matched entities;
- VT is the percentage of very trustworthy matched entities(defined in relation to the size of the smallest of the two VT class, i.e. 313 members);
- Err is the average position error, i.e. the average of the distance D between the two classes for all the entities in the population: $Err = avg(D)$.

The table IV summarises the results for each metric, this using our searching heuristic. This illustrates the 4 parameters that maximised the metric, the optimal, the average of the metric, the worst percentage of the prediction gained by our adaptation in comparison to the average. Note how for Err the problem is a minimisation one.

Metric	W_a	W_p	W_f	W_r	Optimal	Avg	Worst	Gain
M	0.89	0.39	0.11	0.01	60.29	54.44	45.7	10.82
VT	0.20	0.79	0.12	0.11	78.30	45.29	9.58	72.94
ERR	0.89	0.79	0.29	0.09	0.57	0.64	0.93	11.13

Table IV
WEIGHTS OF EACH TRUST SCHEME THAT MAXIMISE THE METRIC.

The preliminary obtained results show a good gain in the quality of the prediction as the result of the adaptation process. The more positive the result is, the better the model predicts (very trustworthy users, with a 78.3% matching). Without adaptation, the system showed a poor average of 45.29% of matching.However, the model seems not valid in predicting other Trust classes, with an overall matching of 60.29%. The average position error of 0.57 means that, on average, the system puts a member in the wrong class half of the time, but this wrong class is the one adjacent to the right one. Finally, the vast majority of matching error is due to the a priori choice of thresholds that implies that the population has an unexpected behaviour that seems to contradict our choices. If in the computation of the matching we consider only the entities that can be actually matched, results are different. The entities that can actually be matched are equal to the sum of the minimum number of entities for each couple of Trust classes. For instance, for the pair of very untrustworthy (VU) Trust classes, the number of matchable entities is the minimum value between 25774 and 15653. The resulting total number of matchable entities for all the classes is 21155. The highest percentage of matched entities (see table IV, metric M, optimal value) was 60.29%, corresponding to 18875 entities. This fact shows that, since $18875/21115 = 0.893$, almost 90% of the population was correctly matched by the adaptive algorithm.

V. CONCLUSION AND FUTURE WORKS

In this paper we addressed the problem of self-adaptation in Trust computations. We presented a methodology to update the Trust function via a training algorithm in order to obtain current and contextualised degree of trustworthiness of virtual identities. The approach proposed is based on a set of Trust schemes which represent the pieces of Trust evidence in a given environment. In this work the author adopted the temporal Trust properties of the LTTM model [5] both for descriptive and for evaluation purposes. In order to model an environment, the methodology firstly proposes a reasoning process among the pieces of Trust evidence, properly validated by a theorem prover, in order to create a set of rules useful to attribute a Trust class for a given entity according to predefined thresholds.

Similarly, a Trust function is initially created by considering a simple equally weighted aggregation of the Trust factors. An adaptive heuristic is proposed to adapt these weights. Finally, a comparison between the entities' Trust class obtained both by the *reasoning-rules based computation* and the *weighted Trust function*, by adopting an a priori sampling distribution, is performed in order to validate the accuracy of the system as a whole. The adaptive algorithm is iterated until the desired confidence level is reached or for a predefined number of iterations, by taking the maximum accuracy output. The evaluation took into account three different metrics in order to test the automatic adaptation of the Trust function. The preliminary results show a good gain in the quality of prediction, even if a boolean threshold has been adopted to produce the Trust classes in the reasoning-rules based computation. This result suggests the methodology is promising and further research may be carried on. The adoption of on-line learning algorithms, with strong mathematical background properties, represent the next step towards the generalisation of the proposed approach and multiple regression techniques would be useful for comparisons.

REFERENCES

[1] L. Priddy K., E. Keller P. *Artificial Neural Network: an Introduction*. Book. Isbn: 0819459879

[2] Seigneur J.M., Jensen. C. *Trading Privacy for Trust*. Proceedings of Trust 2004, pp. 93-107, 2004.

[3] Friedman E. J., Resnick P. *The Social Cost of Cheap Pseudonyms*. Journal of Economics and Management Strategy 10(2): 173-199, 1999.

[4] Cahill V., et al. *Using Trust for Secure Collaboration in Uncertain Environments*. IEEE Pervasive Computing Magazine, vol. 2, N. 3, Special Issue July-September 2003.

[5] Longo L., Dondio P., Barrett S. *Temporal Factors to evaluate trustworthiness of virtual identities*. Third International Conference on security and privacy in communication networks, pages 11-19. IEEE, SECURECOMM 2007. Nice, France

[6] Marsh S. *Formalizing Trust as Computational Concept*. PhD thesis, University of Stirling, 1994.

[7] Gorobets. A. *Agent based Computational Model of Trust*, Erim Report Series, July 2006.

[8] Quercia D. *STRUDEL: Supporting Trust in the Establishment of Peering Coalitions*. ACM SAC06, pp. 1870-1874, 2006.

[9] Luhmann N. *Familiarity, Confidence, Trust: Problems and Alternatives*. Chapter from the book Trust: Making and Breaking Cooperative Relations, pp. 213-237, 2000.

[10] www.finanzaonline.it Last accessed on the 10th, June 2009.

[11] Romano D. *The Nature of Trust*, Louisiana University. PhD Thesis, 2003.

[12] Wang Y., *A Bayesian P2p Trust Model*. AP2PC03, 2003.

[13] Dondio P., *Trust as a Form of Defeasible Reasoning*. Phd Thesis, Trinity College Dublin, 2008.