Reputation Based Trust Management System Supporting Collaboration in a Medical Application

Roman Špánek[1,2], Daniel Kouřil[3,4,5],
Martin Kuba[3,4,5], and Michal Procházka[3,4,5]

[1] Technical University of Liberec, Czech Republic
[2] Institute of Computer Science, Czech Academy of Sciences, Prague, Czech Republic
[3] Institute of Computer Science, Masaryk University, Brno, Czech Republic
[4] CESNET, Prague, Czech Republic
[5] Faculty of Informatics Masaryk University, Brno, Czech Republic
roman.spanek@tul.cz
http://www.cs.cas.cz/spanek/

Abstract. In a small group of people it is quite easy to start a collaboration based on shared trust, because people quickly recognize quality of each other. But this is not true when we move to the highly distributed environment with hundreds of users, not only from one institution or town, but even from different countries. It becomes very complicated task to distinguish experienced an trusted people from malicious users.

In the paper a reputation system proposed particularly for a medical environment supporting collaboration among physicians is presented. The system provides tools for sharing knowledge and expertize in form of modules which can be seamlessly connected to each other provided more advanced functionality. The reputation system is designed to help physicians selecting the right (the most reliable) module from set of all modules within the system. The selection is made on trust being managed by the proposed reputation system.

Keywords: Trust, Reputation System, Security.

1 Introduction

People can usually achieve particular goal easier and faster if they share knowledge and experience. From that point of view sharing information is crucial for every contemporary research discipline. However, the dark side of information sharing is a variable quality of results since results based on inaccurate or faulty inputs are not valuable. Each piece of information that comes as input should be properly assessed as to its quality.

A possible way of information evaluation is to assess its originator. When seeking a particular piece of knowledge, i.e., a suggested procedure to solve a problem, people tend to accept information coming from a trusted person/source, who is known to be familiar (an expert) with the problem.

J.J. Yonazi et al. (Eds.): ICeND 2011, CCIS 171, pp. 285–297, 2011.

Such approach can be quite easily applied in a closed environment consisting only of several people. However, it is much more difficult to mount the same mechanism in an open and distributed environment where people have even not met each other, but still want to collaborate on the same topic.

In this paper we present a reputation system designed for a MediGrid environment designed to support collaboration between experts in BioMedicine.

2 Knowledge Sharing in MediGrid

The goal of the MediGrid project [11,12] is to provide a platform supporting sharing knowledge in the biomedical domain. The knowledge that can be handled by the system must be expressed algorithmically and is inserted into the system as a *module*. Each user using the MediGrid system can use modules produced by others and they can also produce their own modules that can be used by other users, i.e. every user of the MediGrid can act both as the producer and the consumer of modules (in this way the MediGrid behaves in the same way as Peer-to-peer network).

Each module in the system has a detailed description of its functionality. Particularly the following data are required to be added to each module:

- list of scientific publications describing the functionality realized by the module,
- description of inputs and outputs (together with data types),
- several examples of input and output data to test the module implementation,
- modules can be assigned with tags, which help users searching for a module providing desired functionality.

Modules are grouped into so called *kits*, which are supposed to collect modules from various parts of medicine, e.g. neurology, urology, pediatrics etc. or modules used in one place, e.g. in an emergency room or in a genetic laboratory. A kit is a kind of working desktop on which user (users) may group modules and eventually connect more modules to provide more sophisticated functionality.

From a user point of view, a module providing desired functionality has to be somehow validated that provided outputs are correct and reliable. This is not an easy task, as MediGrid is aimed to be totally distributed environment. Solving this issue by traditional security systems (like PKI, passwords, etc.) is not sufficient in such environment.

Situation is getting even more complicated as every registered user can insert her own modules, can access modules of other registered users and create her own kits. In order to simplify the decision, a reputation system was designed with respect to all peculiarities of the MediGrid system. The main purpose of the designed reputation system is to assign a reputation to each module in the system. The score is then used for ordering modules from the most reputable users (note that all modules fulfilling filtering criteria are listed).

3 Calculation of a Module Reputation

Reputation system is designed to accept several variables as an input and to produce a real number output representing reputation of selected module, kit or author. In the following we describe basic points influencing the reputation systems design together with definition of a module, user and kit reputation.

3.1 Reputation System Design

The design requirements were directly influenced by the MediGrid environment peculiarities. As was already mentioned, the MediGrid is aimed to be a system based on Computing Grid ideas proving an extensive support to knowledge sharing within Biomedicine environment. The main target - knowledge sharing - should be supported by a very flexible, open and easy to use solution as users are experts in biomedicine,but may be af limited knowledge in computer science.

The following list summarizes set of requirements being observed when designing reputation of a module.

- Reputation of a module has to be based on:
  - reputation of the module author,
  - reputation of a kit the module is placed on,
  - popularity of the module (how many times the module has been used).
- The output of the reputation system should be simple and easily understandable by regular users.
- users should have a possibility how to evaluate quality of modules manually (by giving a positive or negative feedback).

The most of requirements are simply based on the proposed MediGrid structure (modules grouped on kits), but some are also motivated by a fact, that end users are highly educated in biomedicine, but have limited experience in computer science. The former observation is particularly important as most of known reputation systems are aimed for users being at least familiar with computers. In the MediGrid this is no longer valid.

3.2 Designing Reputation Calculation

The idea of the algorithm is that the reputation of a module depends on several components (mentioned in the previous subsection 3.1).

In order to give users rating of selected module as simple as possible, the five stars rating scale was used. Such rating are very often used and easily acceptable by most of common users.

The following list gives a basic informal definition of reputations:

- **Module**: The reputation of a module is calculated from
  1. its own rating given by users on the scale from one to five stars,
  2. on the number of times the module has been used, and
  3. on reputation of its author.

- **Author**: The reputation of the author depends on
  1. the reputation of all the modules the user has created,
  2. the reputation of the kits the user has created.
- **Kit** The reputation of kits depends on:
  1. the reputation of modules on a given kit,
  2. the number of times the kit has been used.

From the definition above it is clear that reputation of a module cannot be computed in one step as the module reputation depends on the reputation of its author; the author reputation uses reputation of kits and also the reputation of the modules used has created. Such situation is graphically depicted in figure 1 and it is well known for example in database systems as a *deadlock*.

Taken literally, such reputation definition would create a circular dependency, as the reputation of a module depends on reputation of its author which depends on reputation of his/her modules. On the other hand, omitting of any of the reputation components (the author reputation, kits reputation) would result in less reliable output. In order to remove the circular dependency, the reputation of a module is computed in three parts, marked a,b and c, where a expresses the average rating, b expresses the number of usages and c expresses the reputation of the author. When computing reputation of an author or a kit, only the parts a and b are used, as the c part cannot be determined in that time.

The algorithm for a given module produces a real number by calling the function `module_reputation(module)`.

The three parts are real numbers between 0 and 1. The final real number is produced by the following formula:

$$reputation_{module} = (a+1)^{(1/wa)} \times (b+1)^{(1/wb)} \times (c+1)^{(1/wc)} - 1 \qquad (1)$$

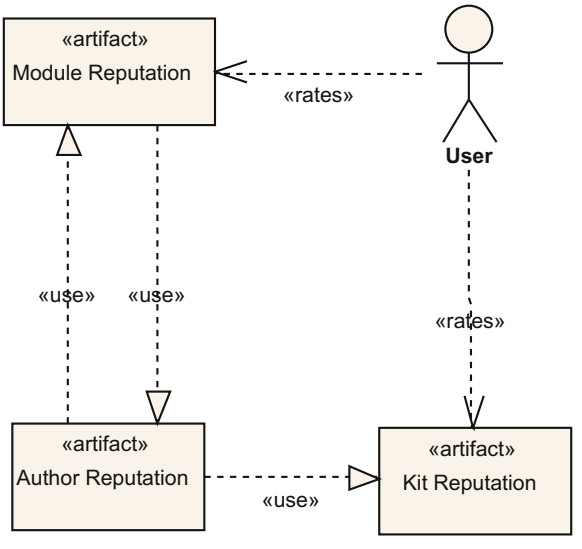

Fig. 1. Diagram depicting dependences between a module, kit and author reputation

The formula uses the following weights wa,wb,wc that have to satisfy the following criterion:

$$\frac{1}{wa} + \frac{1}{wb} + \frac{1}{wc} = 1 \tag{2}$$

Similarly, the reputation of a user (author) is computed in two steps marked d and e, where d expresses reputation of the user's kits and e expresses reputation of all of the user's modules. The parts are subsequently weighted by using the following formula:

$$reputation_{author} = c = (d+1)^{(1/wd)} \times (b+1)^{(1/we)} - 1 \tag{3}$$

where weights hold the following condition

$$\frac{1}{wd} + \frac{1}{we} = 1 \tag{4}$$

The reputation of a kit is computed in the same fashion. Particularly, by dividing the total reputation into two parts g and h, where g expresses the number of times the kit has been used and h expresses the reputation of all the modules on the kit. The parts are also weighted using the following formula

$$reputation_{kit} = (g+1)^{(1/wg)} \times (h+1)^{(1/wh)} - 1 \tag{5}$$

where proposed weights wg and wh satisfy the following condition:

$$\frac{1}{wg} + \frac{1}{wh} = 1 \tag{6}$$

Introduction of the weights was necessary for the fact that scales of the reputation components are different. The given formulas require parameters a, b, c, d, e, g, h to be on 0 and 1 scale and the formulas produce a number on the same 0 and 1 scale. Parameters c, d, e, h are products of weighting or averages of weightings being also of 0 to 1 interval. The other parameters are computed as simple ratios:

- a is the rating divided by the length of rating scale,
- b is the number of module usages divided by the number of all usages of modules being on the same kits as the given module (note that a module can be possibly used on many kits simultaneously),
- g is the number of kit usage divided by the number of all usages of all modules.

The weights used in the reputation components can be set as a constant. On the other hand, it is required to give more importance to a rating provided by more users (simply larger group of users has been determined to give a same/similar either positive or negative rating). Thus weight wa is computed with respect to the overall number of users providing the rating feedback.

Weights wc, wd, we, wg, wh are global constants, and each module is given by its own weights wa, wb.

Calculating components a, b of reputation of authors and kits cannot use the same weights wa, wb, so another pair wam, wbm is introduced fulfilling the following criteria:

$$\frac{1}{wam} + \frac{1}{wbm} = 1 \tag{7}$$

The algorithm uses the following functions for obtaining primary data from the MediGrid:

- `author(module)` returns user who is the author of the module given as the parameter
- `created_kits(user)` returns a set of kits that the user created
- `created_modules(user)` returns set of modules the user created
- `modules_in_kit(kit)` returns set of modules in a given kit
- `used_num(module)` returns int, the number how many times the module has been used by other users (omitting the module author)
- `used_num(kit)` returns int, number expressing how many times the kit has been used (omitting the kit author)
- `all_usages()` returns int, number expressing overall number of module usage, each usage of a module in a kit
- `usages_similar_modules(module)` returns int, total number expressing how many times modules in the same kits have been used (including the module the function is called for)
- `all_rating_users_num()` returns int, total number of all users who put any feed back on any module in the system
- `rating_users_num(module)` returns int, number of users who have given a feedback for a given module
- `last_user_ratings(module)` returns set of tuples [user,rating], set of ratings of the module from all users who has given a feedback (except for the module's author). the function returns only the latest rating for each user.

The following pseudo code describes the functionality and it fives several implementation details.

```
#the final module reputation
function module_reputation(module) returns real
   [a,b] = module_reputation_a_b(module)
   c = scalarize_user_rep(user_reputation(author(module)))
   return scalarize_module_rep(module,a,b,c)
end function

#the parts A and B of a module reputation
function module_reputation_a_b(module) returns [real,real]
   a = rating(module)/5
   if usages_similar_modules(module)==0 then
     b = 0
   else
     b = used_num(module)/usages_similar_modules(module)
   end if
```

```
   return [a,b]
end function

function compute_weights(module) returns [real,real,real,real]
   wa = all_rating_users_num() / rating_users_num(module)
   wb = 1 / (1 - 1/wa - 1/wc)
   wam = wa
   wbm = 1 / (1 - 1/wam)
   return [wa,wb,wam,wbm]
end function

# average evaluation of the module from all users
function rating(module) returns real
 sum = 0
 num = 0
 forall eval in last_user_ratings(module) do
     sum = sum + eval.rating
     num = num + 1
 end for
 if num==0 then return 3 else return (sum/num) end if
end function

#user reputation in three parts
function  user_reputation(user) returns [real,real]
   d_sum = 0
   d_num = 0
   forall kit in created_kits(user) do
      d_sum = d_sum + scalarize_kit_rep(kit_reputation(kit))
      d_num = d_num + 1
   end for
   if d_num == 0 then
      d = 0
   else
      d = d_sum / d_num
   end if
   e_sum = 0
   e_num = 0
   forall module in created_modules(user) do
      e_sum = e_sum + scalarize_module_a_b(module,module_reputation_a_b
      (module))
      e_num = e_num + 1
   end for
   e = e_sum / e_num
   return [d,e]
end function

function kit_reputation(kit) returns [real,real]
   g = used_num(kit)/all_usages()
   h_sum = 0
   h_num = 0
```

```
   forall module in modules_in_kit(kit) do
     h_sum = h_sum + scalarize_module_a_b(module_reputation_a_b(module))
     h_num = h_num + 1
   end for
   h = h_sum / h_num
   return [g,h]
end function

function scalarize_module_rep(module, [a,b,c] ) returns real
   [wa,wb,wam,wbm] = compute_weights(module)
   return (a+1)^(1/wa) * (b+1)^(1/wb) * (c+1)^(1/wc) - 1
end function

function scalarize_module_a_b( [a,b] ) returns real
   [wa,wb,wam,wbm] = compute_weights(module)
   return (a+1)^(1/wam) * (b+1)^(1/wbm) - 1
end function

function scalarize_user_rep( [d,e]) returns real
   return (d+1)^(1/wd) * (e+1)^(1/we) - 1
end function

function scalarize_kit_rep( [g,h] ) returns real
   return (g+1)^(1/wg) * (h+1)^(1/wh) - 1
end function
```

4 Experiments

MediGrid system is a real application being currently used by a group of biomedicine experts. The proposed reputation system has been implemented in Java accepting required data coming from a relational database (data about usage, ratings etc.) and storing/modifying reputation of modules, kits and authors into the same database. MediGrid web interface then accesses stored data and produces a set of screens giving user lists of modules, kits and also users with additional information about their reputation (see figure reffig:medigridexample).

The main aim of the experiments was to experimentally verified the weights and also parameters in the reputation formula are set properly and also to verify their influence on the reputation system functionality.

The experiments have to be run in background, as MediGrid is used by biomedicine experts so any unexpected behavior might be very unpleasant causing unwillingness to use MediGrid. Therefore the experiment was setup in the following manner:

- Four new accounts had been created and assigned to four different users (coauthors of this paper).
- Each user has been given by a role in the system:

- a proper user, producing modules of high quality,
- a standard user producing few modules of limited quality mostly consuming provided functionality,
- a fake user, trying get better reputation by copying modules, creating kits from highly rated modules etc.

For the fact that none of coauthors is an experts in the biomedicine area, the experiments have to be limited to just this group of users (new modules were created from existing modules in the system based on the global knowledge of expertize of users in the MediGrid).

The experiments have shown, that weight has strong impact on the reputation of modules (kits and users). As a result, tho following weights has been identified as the most reliable configuration $wc = 4$ $wd = 2$ $we = 2$ $wg = 1.5$ $wh = 3$.

In figure 2 is shown an example listing of modules with calculated reputation (depicted in a five start scale). The listing also shows values of each component (this is mainly for experimental purposes).

5 Related Work

5.1 Trust Management

Policy based approach has been proposed in the context of open and distributed services architectures [2],[3] as well as in the context of Grids [1] as the solution to the problem of authorization and access control in open systems. Its focus is on the trust management mechanisms employing different policy languages and engines for specifying and reasoning on rules for the trust establishment. Since the primary aim of such systems is to enable access control, trust management is limited to verification of credentials and restricting access to resources according to policies defined by required resources owner [6].

On the contrary, *Reputation based* trust management systems provide a way in which entities may evaluate and build a trust relationship between resource provider and requester. Reputation approach emerged in the context of electronic commerce systems, e.g. eBay. In distributed settings, reputation-based approaches have been proposed for managing trust in public key certificates, P2P systems XREP, mobile ad-hoc networks, and recently, also in the Semantic Web [4], NICE [13], DCRC/CORC [7], EigenTrust [9], EigenRep[10].

Social network based trust management systems utilize, in addition, social relationships between entities. In particular, the social network based system views the whole structure as a social network with relationships defined amongst entities. Examples of such trust management systems include Regret [17], NodeRanking [16].

5.2 Dynamic Trust Management

Trust is in some application a static phenomena, but in many other environments (e.g. Peer-To-Peer networks (P2P), mobile databases, the semantic web, the real human society) trust is highly dynamic.

Reputation overview

Main page
Users
Modules
Indicator classes
Citations
Reputation

POWERED BY Stripes
Log in

Weights

a - average rating from users
b - ratio of this module usages to usages of all modules belonging to same kits
c - reputation of module author
d - average reputation of user's kits
e - average reputation of user's modules, computed from a and b parts
g - ration of this kit usages to all kit usages
h - average reputation of modules in a kit, computed from a and b parts
module reputation $rep_{mod} = (a+1)^{(1/wa)} * (b+1)^{(1/wb)} * (c+1)^{(1/wc)} - 1$

Modules

id	module name	rep_{mod}	wa,wb	a	b	c	Author	d	e
34	Body Mass Index	0.46	2, 4	0.93 ★★★★½	0.1613 (20/124)	0.04	Jana Smržová	0.05	0.04
43	Estimate of total body water as % of body mass	0.29	3, 2.4	0.9 ★★★★½	0.0698 (9/129)	0.04	Jana Smržová	0.05	0.04
35	Estimate of ideal body weight	0.26	3, 2.4	0.7 ★★★½☆	0.1165 (12/103)	0.04	Jana Smržová	0.05	0.04
31	Free water clearance	0.25	∞, 1.33	-	0.3333 (3/9)	0.04	Jana Smržová	0.05	0.04
93	coversion from cm to feet and inches	0.21	6, 1.71	0.8 ★★★★☆	0.1429 (3/21)	0.05	Martin Kuba	0.05	0.06
101	Romanuv Body Mass Index	0.19	∞, 1.33	-	0.2069 (6/29)	0.13	Roman Spanek	0.11	0.15
104	Předpokládaná výška dítěte	0.17	6, 1.71	1 ★★★★★	0.0286 (3/105)	0.1	Michal Prochazka	0.08	0.13
92	převod °F na °C	0.15	∞, 1.33	-	0.1905 (4/21)	0.05	Martin Kuba	0.05	0.06
98	Lepsi dan z prijmu	0.15	6, 1.71	0.2 ★☆☆☆☆	0.1538 (10/65)	0.12	Daniel Kouril	0.09	0.16
39	Odhad celkové tělesné vody (total body water) podle Watsona	0.13	6, 1.71	0.8 ★★★★☆	0.0175 (1/57)	0.04	Jana Smržová	0.05	0.04

Fig. 2. Example listing of modules in the MediGrid system with calculated reputation

In most dynamic approaches trust is defined as a vector comprising few factors contributing to the overall trust value (e.g. [5]):

- the *short term trust factor*,
- the *long term trust factor*,
- the *penalty factor*.

These factors are then combined into one value of *dynamic trust metric* of a particular connection between entities. The purpose of the factors can be generalized as an effort to accommodate sudden deviation in normal behavior of an entity (so-called oscillation) together with long term behavior observation. The penalty factor is concerned to make reaction of the system (decrease or increase of trust level) satisfactory.

5.3 Trust Management System Standardization

Many systems have been designed without any standardization. In other words, trust management systems have been proposed and designed particularly for a target environment. Such situation was identified as a possible issue and several works have tried to identify some common parts [15], [14]. Such studies are very important as it is shown that despite differences in for example computational models, the same main building blocks can be found in each or at least majority of trust management systems. As trust management systems using reputation as the basis have been studied in the first place we present the common parts selected for such systems [8]:

- gathering behavioral information,
- scoring and ranking entities (peers, nodes, agents, and sensors),
- entity selection,
- transaction,
- and rewarding and punishing entities.

6 Conclusion

Paper describes a reputation system designed particularly for MediGrid environment that is aimed to be an open infrastructure based on computational grids supporting collaboration and knowledge sharing between experts in biomedicine. The reputation system design was driven by the fact that users in such open infrastructure lack of decision support which module/part is more reliable and will provide better service with higher probability.

The reputation system was shown to be proper solution providing users, who are not computer experts, with a simple and straightforward way how to find out which modules, kits and authors are the most reliable/trustable.

Reputation system was successfully implemented as a new module into the MediGrid environment. The experiments have shown that such solution is well accepted by the end users.

Acknowledgments

This work was realized under the state subsidy of the Czech Republic within the research and development project "Advanced Remediation Technologies and Processes Center" 1M0554 - Programme of Research Centers supported by Ministry of Education

References

1. Basney, J., Nejdl, W., Olmedilla, D., Welch, V., Winslett, M.: Negotiating trust on the grid. In: 2nd WWW Workshop on Semantics in P2P and Grid Computing, New York, USA (May 2004)
2. Becker, M.Y., Sewell, P.: Cassandra: distributed access control policies with tunable expressiveness. In: 5th IEEE International Workshop on Policies for Distributed Systems and Networks, Yorktown Heights (June 2004)
3. Bonatti, P.A., Olmedilla, D.: Driving and monitoring provisional trust negotiation with metapolicies. In: 6th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2005), pp. 14–23. IEEE Computer Society, Stockholm (2005)
4. Damiani, E., di Vimercati, S.D.C., Paraboschi, S., Samarati, P., Violante, F.: A reputation-based approach for choosing reliable resources in peer-to-peer networks. In: Proceedings of ACM Conference on Computer and Communications Security, pp. 202–216 (2002)
5. Duma, C., Shahmehri, N., Caronni, G.: Dynamic trust metrics for peer-topeer systems. In: Proceedings of 2nd IEEE Workshop on P2P Data Management, Security and Trust (in connection with DEXA) (August 2005)
6. Grandison, T., Sloman, M.: Survey of trust in internet applications. IEEE Communications Surveys 3(4) (2000)
7. Gupta, M., Judge, P., et al.: A reputation system for peer-to-peer networks. In: Thirteenth ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video, Monterey, California (2003)
8. Hoffman, K., Zage, D., Rotaru, C.N.: A survey of attack and defense techniques for reputation systems. ACM Computing Surveys 42, 1–31 (2009)
9. Kamvar, S., Schlosser, M., et al.: The eigentrust algorithm for reputation management in p2p networks. In: WWW, Budapest, Hungary (2003)
10. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: Eigenrep: Reputation management in p2p networks. In: Proceedings of 12th International WWW Conference, pp. 640–651 (2003)
11. Kuba, M., Krajíček, O., Lesný, P., Vejvalka, J., Holeček, T.: Grid empowered sharing of medical expertise. In: Proceedings of HealthGrid 2006, pp. 273–282 (2006)
12. Kuba, M., Sebestianová, Z.: Web deployed interface for a medical knowledge grid. In: The Second International Conference on Internet and Web Applications and Services, ICIW 2007 (2007)
13. Lee, S., Sherwood, R., et al.: Cooperative peer groups in nice. In: IEEE Infocom, San Francisco, USA (2003)
14. Mekouar, L., Iraqi, Y., Boutaba, R.: Reputation-based trust management in peer-to-peer systems: Taxonomy and anatomy. In: Handbook of Peer-to-Peer Networking: Part 6, pp. 689–732. Springer Science + Business Media, Heidelberg (2010)

15. Mrmol, F.G., Pérez, G.M.: Towards pre-standardization of trust and reputation models for distributed and heterogeneous systems. Computer Standards Interfaces 32, 185–196 (2010)
16. Pujol, J., Sanguesa, R., et al.: Extracting reputation in multi agent systems by means of social network topology. In: First International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy (2002)
17. Sabater, J., Sierra, C.: Regret: A reputation model for gregarious societies. In: 4th Workshop on Deception, Fraud and Trust in Agetn Societies, Montreal, Canada (2001)