Improving Recommender Systems Using Knowledge Obsolescence as a Predictor of Trust

Pablo Cruz Navea
Master of Science in Informatic Engineering
Departamento de Informática, Universidad Técnica Federico Santa María
Valparaíso, Chile
Web: http://about.me/pablocruznavea

***Abstract*—In the current context of the Social Web, trust has emerged as a concept and mechanism to differentiate users of this Social Web and the content they generate. Much effort has been devoted to study trust predictors with the aim to provide some operational use of the concept. We propose in this work a new predictor for trust: knowledge obsolescence. We provide a characterization of the concept and a description of the relation between trust and knowledge obsolescence. We applied the concept to a generic recommender system. For this purpose, we have developed a software simulator that allow us to test trust and knowledge obsolescence networks in the recommender systems context. Interesting results were obtained. We found that recommender systems success is augmented. Moreover, we found an improvement in some cases for the coverage of potential recommendable items. We did not find statistical significant benefit on the quality of recommendations.**

***Index Terms*—Recommender systems, trust-based recommenders, knowledge obsolescence, trust models.**

I. Introduction

The web is changing. From the old web sites where content was mainly author-of-the-site generated, we are now experiencing the appearance of new web applications where the content is also user-generated. This is a huge paradigm shift in which users are now not only consumers, but also producers of content.

Internet has demonstrated being a great enabler for virtual communities. Social networks, photo communities among other Web applications are now running and experiencing this new paradigm. Users can now share comments, opinions, ratings, thoughts, knowledge, among other social assets.

Several users generating contents imply vast amounts of data. And this imply users that want to consume some specific data will face problems to find the data they really want. Moreover, not all users generate data with similar quality. Thus, in the need to differentiate users and content they generate, we need to incorporate some elements to accomplish this. One key element is *trust*.

Several effort in incorporating trust in web applications have been made. One example is trust-based recommender systems. Recommender systems appeared as web applications with a clear goal: to facilitate users in the search and find of specific items (movies, camcorders, etc.) or data. With the inclusion of trust, recommender systems have shown improvements in quality of items or data recommended, coverage of items or data and alleviating some inherent problems such as the cold-start problem.

Although discussed several years ago in [1], knowledge obsolescence has not been considered as predictor of trust, and thus the concept has not been used yet to improve recommender systems. In this thesis work, we propose a trust model which considers knowledge obsolescence as a predictor of trust. We present a characterization of the knowledge obsolescence concept appropriate for operational and computational use and then we use this inferred trust value to improve a generic trust-based recommender system.

A software simulator was developed and is presented in this work. We simulated several trust-network scenarios in order to find where knowledge obsolescence improves recommendations. We present a validation of the simulator and the analysis of the results given by the simulator.

This work is structured as follows. Section II presents the context of this work and the problem we found interesting to be worked on: issues with operational uses of trust in Web applications and the reasons that motivated us to consider knowledge obsolescence. Section III presents related previous work. Section IV presents the main proposal of this work: (1) a trust model and (2) a characterization of the knowledge obsolescence concept. Sections V and VI present an implementation of our proposal and its validation. Section VII presents results given by several simulations runs and their analysis. Finally, section VIII presents the conclusions of this thesis work and some interesting challenges and ideas for future work.

II. Context and related problem

A. Context

Social network concept is commonly heard today regarding interaction between persons, especially on Internet, but it is important to mention that Barnes introduced in 1954 the first definition of social network [3] describing them as a system in which ties between pairs of persons who regard each other as approximate social equals exist [4]. Although Barnes makes no use of mathematical or computational definitions, his statement is applicable to more computational-oriented definitions. In this sense, what Barnes named as *ties between pairs* is what today is conceived as a relationship between users in a social application on Internet (e.g., friendship). And, in more mathematical terms, these relationships are edges in a graph that represent a social network in which vertex or nodes are users (or, in general, any kind of social actor [3][41][69]).

Most, if not all, of the social applications have in common the following aspects:

- Users of the social applications can interact with other users by sharing knowledge, experiences, ideas and opinions, and
- Users use the data in social applications (both, user-generated and author-generated data) for making faster decisions (e.g., buying a product).

In practical terms, *making faster decision* implies some help by means of a recommendation to the user. Several recommender systems in different areas have been implemented. For instance, we can mention a Music Recommendation System that provides a personalized service of music recommendation [80], a recommender system for Web services that aims to help the user picking the service that match his requirements [81], a system to recommend courses in a university [82], a recommender for commodity e-commerce-type products [83] and even for recommendation of photo-shooting points [84]. These examples of recommender systems show that there is almost no limit to which area a recommender system could exist with the same goal: helping users to make faster (and better) decisions.

Trust has emerged as a concept and mechanism to be considered to improve classic recommender systems.

B. Related Problem

We have stated before in this work that Internet and the Web are experiencing a paradigm shift in which users are not acting only as consumers, but also as producers. Having several users producing content may sound great, but, at the same time, as more and more content is generated, the task of finding high quality data is becoming a hard task.

Recommender systems appeared as a result of important scientific and engineering efforts to provide data of interest to users who make use of Web applications and, at the same time, to improve data coverage in search operations. The reader should note that here the word "data" points to any kind of *returnable item* (e.g., movies, books, profiles, and so on). Colloquially, we could say recommender systems try to bring to users not only the right data, but also the right amount of data.

A *second-order* improvement to recommender systems is achieved, in some cases, by incorporating "trust". Using trust in recommender systems require an operationalization of the concept, a task that is not simple as trust is a very complex concept. This operationalization is required because recommender systems are actually computer programs that operate with numbers. Thus, every trust-based recommender system require a set of clearly-defined parameters in order to make trust-based recommendations.

Some efforts have been made to alleviate this problem (see section III). We can find models to define trust as a function of reputation, as an inferred value given some specific factors, among other approaches.

By exploring several Web applications (mostly virtual communities) we have found, in some specific areas, knowledge obsolescence being an important indicator that users consider implicitly to establish a trust relationship to other users. Our own previous work intended to find a relation between trust and knowledge obsolescence [2]. We found interesting relations between trust and knowledge obsolescence in four computer science topics.

As of this writing, neither recommender systems nor trust models have considered knowledge obsolescence to improve recommendations and/or to infer trust, respectively. This poses the following question: why not use knowledge obsolescence as (1) a predictor (or in more general terms, as an indicator) for trust and (2) to improve trust-based recommender systems.

This work thesis aims to define a trust model which allows inferring trust by using knowledge obsolescence and using the inferred value for recommender systems. Also we make use of *in silico* experimentation in order to determine the impact of incorporating knowledge obsolescence in a typical trust-based recommender system. Section IV presents this trust model and discusses some interesting results on the relation between trust and knowledge obsolescence.

III. State Of The Art

A. Trust and Confidence: Social Concepts

A state of the art in trust-based computing cannot begin without stating that trust is a very complex concept. Moreover, it is not only in the computer science domain in which trust has been recognized, formalized and used. Trust has also been considered as an important concept in fields such as military organization [5], economy and market operations [7][12] and human development [13], corporate-consumers relationships [16], social exchange related theories [17], organization of virtual teams [9][43][76], politics [18][19], automation [20][21][22], among other domains.

Although we experience trust and its manifestations every day of our lives, giving a definition and characterization is a great challenge [23]. The complexity of trust as a concept is mainly due to the several application domains [24] (it is almost a natural implication the fact that each of these domains define and use the concept in different ways), a lack of generic representation of trust [25][26], and unclear distinctions in important related concepts such as trust and cooperation [30]. In more operational terms, trust is also affected by the culture in which it is considered [8]. However, the good part in this problem is that, despite the complexity of the concept, it appears there exist some agreement in the literature regarding the importance of trust [5][7][29][70][93], with some authors going beyond by questioning why trust should not be used in social systems if it is so pervasive in them [42].

Although not clear as we would want, there are some efforts in making a distinction between trust and confidence. Luhmann asserts that both concepts refer to expectations (with possibility of deception), but in the case of trust there exist some compromise between the truster and the trustee expressed as risk and options, that is, a person trusts in someone or something with the assumption that there is some risk in the trustee acting wrong (i.e., not acting in the expected manner) and that there exist other options for the truster [14]. In the case of confidence, there is little, if none, possibility of deception (i.e., very low risk) and even if there is a possibility of deception, the confident person has no choices as in the case of a trust relation because the confident does not know what to do as an alternative [14]. According to this point of view, an example of confidence relation appears between airlines and passengers. Passengers often have to make use of airplanes given the fact there is confidence in the plane reaching the destiny and because there are no alternatives (imagine a place a passenger can reach only by plane). A slight different, but compatible, approach in distinguishing trust and confidence is taken by Tonkiss who states that trust presupposes more risk because a person trusts another one by not observing hard facts such as performance and reputation [7] (e.g., trusting in a person based on how the person dresses). For Tonkiss, a confidence relation appears when having hard facts or data (e.g., investing in a good-performance enterprise). A key idea in distinguishing trust and confidence is that there is no clear boundary between both concepts, thus allowing some relation to be in part confidence and in part trust [14]. Adams also agrees that trust requires risk and adds that confidence requires less abstraction because confidence could be formed by observing probabilities [5], that is, some kind of certainty in a prediction [6]. For Adams, pure trust appears in the absence of certainty [6].

Interactions (relations) between individuals in a social system are fundamental and common [35][36] and are sometimes understood as two-sided and mutually rewarding processes (not necessarily economical) [28]. Emmerson suggests three aspects subject to empirical observation in order to understand social relations: actions by individuals (as noted by [29], some kinds of *webs of reputation* have been observed in settlements as elements acting to control people's behavior), transactions between individuals, and exchange relations (a series of transactions between same individuals) [28]. It is interesting to see how trust and related concepts appear recurrently in the literature and it is more impressive that the concept appears independently of the the social system enabler (i.e., it does not depend on technology enabling a social network). In fact, roots of social capital[1] theory state that networks of relationships between people conform the basis for trust [87]. Regarding social capital, the authors in [10] cite several benefits of high levels of trust: social solidarity, cohesion, better economic performance, support for democracy, and control of immoral behavior. Although these benefits have been observed outside a possible *social capital on the Internet*, it is clear that today, social solidarity, cohesion and control of immoral behavior are key aspects in social networks on the Internet. One can easily argue that an Internet social network with any (or all) of these aspects absent would probably lead to a failing social network application. This aspect of trust in social networks has not only been observed in Internet applications, it also affects the way people get involved with real (i.e., no virtual) institutions [88] (social networks also play a role in behavior styles diffusion [89]). It has also been shown that social capital is critical in knowledge sharing in social networks or, in more general terms, in virtual communities [11]. It is important here to note that social actors do not transfer knowledge and other assets without evaluating quality. This is a key characteristic that make sense of trust in social systems and social capital.

B. Social Networks and Trust

Although initially [3] coined as *system of ties between pairs of persons who regard each other as approximate social equals* [4], the concept of social network brings together the concepts of social system, social interaction, social capital and social exchange. It is important to note that although Barnes analyzed the social organization of Bremnes (Norway) and described a social network in terms of the particular social organization in Bremnes, the concept is compatible with what today we understand as a social network in several areas, including computer science and Internet. It is key to understand social networks the fact that relationships between social actors are necessary to allow the flow of resources [37]. Facebook is probably the best example of a social network enabled by Internet. In fact, Facebook has allowed several studies [33][34] regarding the "small-world phenomena" [31][32].

The emergence of social networks on Internet and, in general, web applications allowing interaction between users (i.e., virtual communities [77][78]) is motivating a paradigm shift [38][79]. From the old Web sites in which Web content was mainly generated by Web site authors, we are now moving to an Internet in which Web content is mainly generated by Web site users [38][39]. Thus, users are now acting as content-authors rather than just content-consumers [40]. Given this paradigm shift, there is an increasing need in establishing mechanisms in order to provide differentiation of the web content that it is generated by the users of web sites (mostly, social applications) [38][40]. This issue is self-evident in the semantic Web in which one fundamental principle is that anyone can produce data [85].

Under the fundamental assumption that individuals do not transfer assets (e.g., knowledge) before evaluating their quality, trust has emerged as one of the elements used to provide differentiation between users and, therefore, for differentiation of the content users generate, such as comments, opinions, reviews, descriptions, thoughts, among others *social assets* (trust can be considered as a key element to facilitate coordination and cooperation [86]).

Some interesting properties of trust can be found in the work done by Sherchan, Nepal and Paris [3]:

- Context specific: trust has a specific scope. This is the case in which, for example, A trusts in B for automobile mechanics, but not for automobile electrical wiring.

[1] Social capital is a very important concept to understand social collaboration and the movement of social assets.

- Dynamic: the trust A has on B can increase, stay equal, or decrease over time.
- Propagative: A can trust on C because there exist *trust links* between A and B and between B and C. Trust can be assumed to decay proportionally to the length of the chain of trust relations [44].
- Non-transitive: trust is propagable, but, in general, trust is not transitive. Transitivity means if A is related to B by some relation R and B is related to C by the same relation R, then A is related to C by the same relation. As an example, if A trusts more in B (than B in A) and B trusts more in C (than C in B), there is no assurance in A trusting more in C (than C in A).
- Composable: there can be several relations between A and C. Therefore, A trusts in C by observing trust in each one of the relations (then composing a value).
- Subjective: it depends on subjective evaluation. Even same behavior could mean different levels of trust for the same person.
- Typically asymmetric: A trusts to a certain extent in B, but there is no assurance the same extent is applied in B trusting A.
- Self-reinforcing: people tend to act positively with others they trust.
- Event-sensitive: even if A has a long-lasting trust relation with B, a single event may destroy trust between A and B.

Another interesting property of trust is *Inferrable*. This property can be observed in the multiple trust models existing today. This property states that trust can be determined by observing other factors. This property is of interest in the computer science domain as this allows trust being computed and used over a trust network.

C. Trust as a Computational Concept: Definitions, Models and Applications

Trust has been studied and applied in different ways in the computer science context and it is rather hard to find consensus on what trust is [42]. Work done by Marsh [94] is commonly cited as the first intent and effort in describing, structuring and formalizing the concept as a computational one. Although in Marsh's work trust is considered as a concept with generic application in the domain, some authors have stressed the idea that trust needs to be considered in interactions between humans [41][96]. In their work, Artz and Gil offer three definitions gathered from previous works by other authors [69]. First definition states that trust is understood as a subjective expectation an agent has about another's future behaviour based on the history of their encounters [69][97]. Second definition stresses the idea that trust is related to the competence of an actor to act dependably, securely, and reliably within a specified context [69][98]. This second definition is more related to the competence to act of an actor rather than the behaviour the actor has shown in the past. Finally, the third definition gathered by Artz and Gil proposes trust as a measurement of the belief that some actor will behave dependably for a specific service offered by the actor within some specified context [69][99]. Despite these three definitions of trust given by their respective authors, Artz and Gil also recognize that trust has sense when it is involved in a relation in which there is a possibility of deception (i.e., deception is a possible outcome) [69]. This is also noted in [14] and [15]. Another view or definition of trust can be found in [91] which states that trust is commonly defined as one having confidence in some service or resource will behave in an expected manner and the authors in [42] go beyond this by including dependability, security and reliability as fundamental characteristics.

Using trust in computer science requires some conceptual framework to define and evaluate trust relationships. The authors of [63] keyed the term *trust management problem* for the definition of security policies, security credentials and trust relationships components in an eventual conceptual framework. This term has evolved since, from a static view to a more dynamic view because it is also recognized that trust changes over time [3][42]. In practical terms, trust management can be considered as the set of principal design decisions (i.e., the architecture of the system [52]) regarding how trust will be computed and stored in the system [45][86]. The reader should note there is some overlapping between what the terms trust management and trust model cover. According to [55], trust management approaches differ in basically three aspects: (1) how trust is computed, (2) the resilience of the architecture to manage trust, and (3) how trust is computed under special circumstances such as in a cold start (people do not only need to trust others, but also to know there are others trusting them [68]). Managing trust in a system includes also non-functional aspects such as reliability and security. Authors of [52] provide some recommendations and guidelines to be incorporated into trust-related architectural styles: use of digital identities, separation of internal and external data, make trust visible, and treat trust as a numerical value which, ideally, would be syntactically and semantically comparable to other values of trust.

In trust management, one can find centralized, distributed and mixed schemes to compute trust. Before delving into some approaches, it is fundamental to remember that Gambetta introduced the word *level* in the definition of trust [68] which is key in the context of computer science as most of the applications of trust in this domain are founded on the idea of working with trust as a level. There are several ways and mechanisms to calculate/derive and propagate trust. MoleTrust [101] and TidalTrust [102] are two widely known algorithms for propagating trust over a network. Although different implementations and designs, both algorithms share the common idea that if one user is connected by a finite number of edges to another user (i.e., there is some way to reach other user by following an existing path in a social network or graph), then the first user could derive a value of trust for the other user (they make use of some of the properties of trust mentioned by [3]). Trust can also be captured or derived from existing data on social applications in a similar way as some reputation systems do [106]. Also, trust can be inferred by using more formal approaches such as applying similarity metrics in a similar way as it is done with graphs [95]. It should be noted that not all approaches are distributed ones. For example, the Advogato project considers a central authority which determines the trust for the users of the application [104]. A kind of mixed-approach is found in EigenTrust in which trust is computed considering local trust values and a global reputation of the peers that rate trust on another specific one [105].

Trust management plays an important role in Medical Information Systems [46][47][48], Information Retrieval Systems [49][51] and Mobile Code [50] (many other generic examples are presented in [42]). Other examples include the following: (1) in crowdsourcing systems, trust is managed to not only find the trustworthy partners, but also to maximize the social welfare of the system (composed by partners) [53]; (2) trust is also managed in mobile and and-hoc networks to encourage nodes behaving correctly and also to evaluate misbehavior [54]; (3) trust management aids in the operation of service provision networks by helping the network to scale for a larger set of individuals, especially in the case of service provision networks with multi-scale service level agreements (SLA) [55]; (4) as the Internet of Things research area matures, security is becoming an important concern [56][57][58][59] and, in this context, trust management is a critical issue to be considered by its very social-network-based nature [60][61]; (5) due to its peer-based nature, a field that cannot be left out of this discussion is P2P systems in which trust management plays a vital role to establish trust among peers [62].

Although trust calculation depends on the specific implementation of an application [64], efforts on characterizing trust as a computational concept are part of *trust models* [42]. *Trust models* generally aim to provide precise definitions of trust [71], the relationships of the attributes that characterize trust and how trust is computed, updated, composed, and measures to counteract malicious users' behavior [65]. Some models define how trust evolves over time [72]. Yan and Holtmanns also make a distinction between trust model and trust modeling, stating that trust modeling is the approach used to represent trust for digital use [86]. Also, parameters being used for inferring trust are commonly studied as part of trust models [69][103]. Trust models are also used for evaluation of *trust-based* systems [73]. Several approaches exist to define trust models: statistical/machine learning, heuristics, behavior-based [3], among others.

Direct Trust is the simplest model with no third-party elements intervening trust computing and management [66]. In this model, a social actor rates trust on a partner directly. Simplicity is a great characteristic of the Direct Trust model, but at the same time this characteristic is less suitable for more complex scenarios such as the case of social networks. A generalization of the Direct Trust model which allows a network made of trust relations is regarded as the Web of Trust model. The Web of Trust model was first coined by Phil Zimmermann in 1992 in the development of the PGP [66][67]. Since then, the Web of Trust model has evolved to more complex variations such as in the case of some recommender systems [100] and in the semantic web [85]. XRep is in part a trust model [52] and part a protocol proposed in [74]. It is, essentially, a reputation-based trust model for P2P file-sharing applications in which a source node chooses a node with some specific file by asking other peers their opinion about the node in question who has been previously selected with a best-match criteria in the initial search for a file. The model was improved by defining new trust semantics, resulting in X^{2}Rep [75].

Literature in computer science domain shows that trust has several practical implications in social applications. Related to this thesis work, it is important to highlight the use of trust for construction of recommender systems. Recommender systems are a relatively old research area in which there is an interest in predicting elements to be offered to an user of an application. When trust is used in recommender systems, they are commonly named as trust-based recommender systems. Trust can be used to alleviate the cold-start problem in recommender systems as it can be used to give more weight to ratings and opinions about some items that could be recommended [111]. Several approaches exist for trust-based recommender systems. TruBeRepec describes an interesting approach of data collection for evaluating trust of users and for using this trust-behavior-based approach in order to provide recommendations of mobile applications [107]. Results from simulations show TruBeRepec not only being an effective recommender system for mobile applications, but also a robust and usable one [107]. In STRS, a social network based recommender system proposed in [108], a trust-reputation based component is used to filter content to achieve higher quality recommendations and to alleviate the cold-start problem (previous results show users will not be rating more items initially in order to obtain more recommendations [109]). Results show improvements in quality and relevance of the recommendations, but also an increase in response time for recommendations as more users are involved (but, being acceptable) [108]. TrustWalker provides another interesting approach in which a two-sided process consists in a random walk in a trust network and a probabilistic item selection [110] with the aim of making a more intelligent walking over the trust network. Experiments with the Epinions dataset show TrustWalker outperforming collaborative filtering and plain trust-based systems in terms of coverage [110]. A slight different approach in using trust in recommender systems (called Top-K recommendation) is found in [114] in which trust is computed by determining the similarity of the interests of users, that is, a kind of similarity-based trust-based recommender system using a modified version of the TrustWalker algorithm [110]. Using two evaluation metrics, experiments show an improvement in quality and performance compared to item-based and user-based collaborative filtering [110]. Trust can also be used to construct recommender systems as explained by O'Donovan and Smyth [112]. This proposal is very interesting for this thesis work. They propose a system which is, actually, a recommender system based on what Resnick et al. proposed in the GroupLens architecture: a recommendation session which consider producers of ratings of items and a consumer to which a rating over a particular item is predicted [115] (i.e., an item that has not been rated yet by the consumer). In its original form, the GroupLens architecture makes use of a matrix involving users (columns) and netnews items (rows) where the elements or entries of the matrix are ratings from users to netnews items. What GroupLens does is to compute or predict a rating for a user that has not rated an item yet. O'Donovan and Symth make use of this approach (heuristic in words of Resnick et al. [115]) recognizing two groups of users that join the *recommendation session*:

- Consumer profile: the user receiving the predicted rating. Strictly speaking, this user is external to the recommendation session.
- Producer profiles: all of the users that has been selected for integrating the recommendation session. Producer profiles have rated other items.

The main idea is one consumer profile gets a predicted rating (the recommended rating by the system) by *requesting the value* to a recommendation session. The recommendation session is formed by producer profiles. When the recommendation session is formed, the predicted rating is computed using 1:

$$c(i) = \bar{c} + \frac{\sum\limits_{p \in P(i)} (p(i) - \bar{p}) sim(c,p)}{\sum\limits_{p \in P_i} |sim(c,p)|} \tag{1}$$

where $sim(c,p)^2$ is a similarity measure which O'Donovan and Smyth replace by the Pearson Correlation [112][3] and:

- $c(i)$ is the predicted rating[4] requested by a consumer c for an item i,
- c is the average of ratings the consumer profile has made,
- $p(i)$ is the rating a producer profile p has made over an item i, and
- $\bar{p}$ is the average of ratings the producer profile p has made over all items.

In the work of O'Donovan and Smyth, there are two options for using trust in the recommender system:

1) Using trust to filter producers. That is, according to a specified trust value used as a threshold, only trustworthy producers are considered in the recommendation session.
2) Using trust to weight ratings according to the level of trust the producer of the rating have.

In case of filtering producers, they simply use the following set of producers:

$$P_i^T = p \in P(i) : Trust(p) > T \tag{2}$$

In the set of producers (equation 2), trust can be profile-level or item-level, as explained by the authors. In simple words, a threshold T is defined. Therefore, only producers with trust greater than T are allowed to participate in the recommendation session.

Using trust to weight rating is a bit more complex. The Resnick's formula is modified as follows:

$$c(i) = \bar{c} + \frac{\sum\limits_{p \in P(i)} (p(i) - \bar{p}) w(c,p)}{\sum\limits_{p \in P_i} |w(c,p)|}$$

where $w(c,p)$ is defined as follows:

$$w(c,p) = \frac{2(sim(c,p))trust(p)}{sim(c,p) + trust(p)}$$

Again, trust can be profile-level or item-level. If item-level trust is used, the weight must consider the trust at item-level by using $trust(p,i)$ and the left part of the equation should be written as $w(c,p,i)$. It should be noted that this weight is obtained by applying the harmonic mean to trust and similarity values (authors argue harmonic mean is more robust in response to large differences between both values [112]).

Results of experiments made by the authors show a positive impact of trust on overall prediction errors [112].

IV. Thesis Proposal and Hypotheses

As stated before in sections I and III, obsolescence is an interesting characteristic of knowledge. The concept applies to a certain degree depending on the knowledge area [1]. We propose here a trust model based on *knowledge obsolescence*. The rationale behind this trust model is founded on the fact that in areas in which knowledge is subject to obsolescence, we expect users being able to rate knowledge obsolescence in a more *natural* way than rating trust (which, in our experience and other reported works, is not simple for a user to rate).

Therefore, we propose trust as a function of knowledge obsolescence, that is, $T = t(k)$, where T is the trust level and t is a generic function that takes as argument the knowledge obsolescence level k. Although we left to the specific application the exact definition of the function $t(k)$, we provide in this section a characterization of knowledge obsolescence based on data gathered from two experiments.

A. Hypotheses

Given the potential applicability of knowledge obsolescence in trust-based recommender systems, it seems reasonable for us to state the following hypotheses:

[2] Consumer profile represent the user is waiting for, or getting a, recommendation. That is, items are recommended to a consumer profile. Producer profiles represent the users that have rated items before (i.e., the ratings producers).

[3] Pearson Correlation is used in the formula to get a correlation between the ratings on items of users in an hypothetical social network. However, the reader should note that using Pearson Correlation is only a specific approach to define "similarity between users".

[4] It is interesting to see how recommendation an item recommendation is mapped to a concrete mathematical operation that predicts a rating for an item that a consumer profile have not rated yet.

- General Hypothesis: Integration of trust and knowledge obsolescence concepts allows improvements in the performance of recommender systems in areas in which knowledge obsolescence has sense.
  - Specific hypothesis: in similar circumstances, trust and knowledge obsolescence provide better quality in results than in a trust-only-based recommender system.
  - Specific hypothesis: in similar circumstances, trust and knowledge obsolescence provide more results (quantity) than the provided ones by a trust-only-based recommender system.

The remaining of this thesis work is devoted to test these hypotheses by using *in silico* experimentation.

B. A Characterization of Knowledge Obsolescence

Understanding what knowledge obsolescence is (i.e., what knowledge obsolescence means in this work) is vital for the development of this thesis. In this context, **knowledge obsolescence** is defined as **a particular state between the highest and the lowest possible currentness of knowledge that a particular social actor has regarding some topic of interest**. That is, (1) given a topic of interest i and (2) a user u who has an opinion regarding a topic i, we define knowledge obsolescence as the **knowledge aging level**, in a continuous range (e.g., from 1 to 10), of the knowledge that the user u has regarding the topic of interest i. As similar as in the case of trust-based social applications, we assume that this aging level is implicit in the opinions or in what user u is sharing with others. This means that some user of an application could determine the knowledge aging level of some other user by reviewing his or her reviews, comments, opinions, among other social assets. It is important to highlight in this context that we assume the aging level is relative to, implicit in, and restricted to the knowledge area. Thus, a specific value for the knowledge obsolescence indicator in one specific area may not (and probably will not) apply to other areas (e.g., knowledge aging level in classical music versus programming languages).

Requiring a user to review what other users have done in the social application has two important practical implications:

1) Social applications considering trust and/or knowledge obsolescence related mechanisms or algorithms need some kind of *user profile*.
2) It is in this *user profile* in which a user would see the *behavior* of the user (in order to rate trust or knowledge obsolescence).

Thus, **knowledge obsolescence is an indicator to express to what extent a particular knowledge is obsolete**. It is important to highlight that a particular knowledge of a person has sense when it is studied regarding some topic of interest. Therefore, the knowledge obsolescence also has sense when studied regarding a topic of interest. It is also important to highlight here that knowledge obsolescence in one topic of interest may not be applicable on other topics. For instance, users that rate knowledge obsolescence on other users regarding the topic *Operating Systems* may not have the same opinion about knowledge obsolescence of the same users regarding a different topic (or knowledge area). This is **a very important issue** and it must be considered when developing and deploying social applications with this concept.

Also, it is important to recognize that not all human knowledge areas are subject to the same. For example, it is difficult to *say* how obsolete is the knowledge of a user regarding classical music which, today, is essentially static. The situation is very different when we consider topics that are more dynamic. For example, software development is a very dynamic topic in which knowledge could get aged in a short time span. Therefore, it is important to recognize the following kinds of topics regarding the dynamic nature of knowledge:

- Static topics: areas in which knowledge is mostly static. The topic or area could be old or new, but the important characteristic is that the knowledge is not subject to high variations through time. This could be the case of classical music which is a knowledge area that is defined and studied regarding musicians and music that was created several years ago (although some small interpretations could vary over time).
- Dynamic topics: areas in which knowledge suffer noticeable variations through time. This is the case of, for instance, software development.

The reader should note that this classification has nothing to do with the *size* of the body of knowledge.

An example of knowledge obsolescence is described in our previous work in [2].

C. Relation Between Knowledge Obsolescence and Trust

After we recognized the idea of knowledge obsolescence and considered it as an indicator for trust, we prepared a pilot study in order to gain some insight about the relation between trust and knowledge obsolescence. This pilot study consisted in asking persons to rate trust and knowledge obsolescence on other persons given four topics of interest: programming of augmented reality systems, system administration in Linux operating system, programming of applications for Android operating system, and object-oriented programming wirh Java or Python.

We considered 16 persons of a Software Architecture undergraduate course at the Universidad Técnica Federico Santa María in Valparaíso, Chile (using students as a source of data is not uncommon as noted in [113]). Each one of the 16 persons were asked to rate trust on, and rate knowledge obsolescence on, between 1 and 10, each classmate in each of the four topics of interest. Students' age range is 22-26 years old.

For obsolescence, 1 means none or very little obsolescence and 10 means very obsolete knowledge. For trust, 1 means none or very little trust and 10 means very high trust.

An important instruction was an indication that if a person could not establish or rate trust or obsolescence (e.g., when a social relationship between the parties has not been established yet) the ratings must be left empty. This is equivalent to say

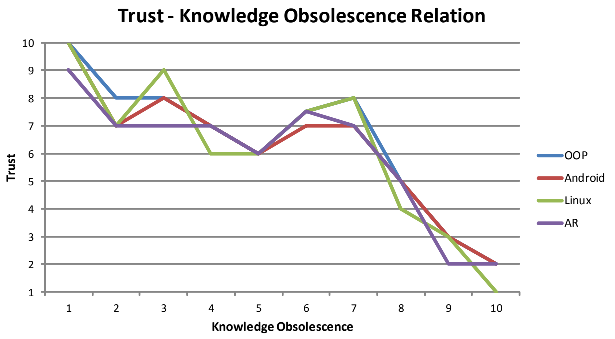

Fig. 1. Trust and Knowledge Obsolescence Relation -pilot study

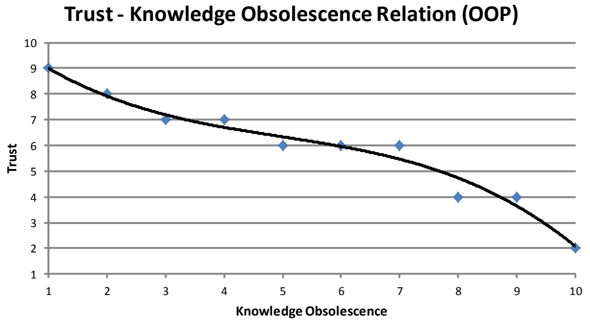

Fig. 2. Trust and Knowledge Obsolescence Relation for Object-oriented Programming - Final study

that the absence of a relationship between two persons is not represented neither by 0 nor 1.

They were also told that the evaluation of their partners has sense when considering opinions or comments about tools or practices in the topics (i.e., the possibility of deception may appear in this information exchange).

Therefore, each subject was asked to rate trust and knowledge obsolescence on their classmates given the following conditions:

- A social relation must exist. It could be virtual (e.g., a virtual transaction on Internet) or real (e.g., they have worked on a project in the past)
- Subjects could rate trust alone, trust along with knowledge obsolescence, or knowledge obsolescence alone
- If a subject could not rate trust or obsolescence (e.g., when a social relationship between the parties has not been established yet) the ratings must be left empty (i.e., neither 0 nor 1)
- They were also told that the evaluation of their classmates has sense when considering opinions, comments or experiences about tools or practices in the topics.

1) Pilot Study Results and Conclusions: From all the valid results entered by the subjects in the pilot study, we obtained 69 pairs of values $\{trust, obsolescence\}$.

As several values of trust were available for each one of the values of obsolescence, the median was used as a grouping measure for the values. The use of the median is motivated by the fact that it is a more robust central-tendency measure than, for example, the simple average and therefore is less susceptible to the influence of outliers.

Two conclusions were drawn from the pilot study:

- Visually, the relation between trust and knowledge obsolescence appeared to be non-linear (see plot in figure 1)
- Augmented Reality and Android Operating System Programming were not widespread topics (only 3 persons rated, in average, another 4 persons).

D. Confirmation Study

Results obtained from the pilot study motivated us to prepare a new study for confirmation. This new study considered only two topics: system administration in Linux operating system and object-oriented programming wirh Java or Python.

Selection of these two topics was guided by the second conclusion of the pilot study [2]. The study applied to 90 persons of a Software Engineering undergraduate course at the same university and in two campuses (Valparaíso and Santiago). Instructions were similar to the given ones in the pilot study. Students' age range is 21-23 years old.

1) Results and Analysis: As in the case of the pilot study, we also used the median here as a grouping measure. But in this case, linear regressions were used for each dataset to explore and describe the relation between trust (T) and knowledge obsolescence (k).

In Object-oriented Programming, the best curve fitted was:

$$T(k) = -0.0198k^3 + 0.2966k^2 - 1.8350k + 10.5667$$

with

- $SSE = 1.1995$
- $R^2 = 0.9692$

In use of Linux Operating System, the best curve fitted was:

$$T(k) = -0.0253k^3 + 0.3561k^3 - 2.0126k + 10.7$$

with

- $SSE = 0.8424$
- $R^2 = 0.9838$

Both equations describe the relations between trust and knowledge obsolescence by using knowledge obsolescence as a predictor variable for trust. Figures 2 and 3 show visually the behavior of trust as as dependent variable on knowledge obsolescence for Object-oriented Programming and Linux, respectively. Additionally, the fitted curves (color black) are shown in each case.

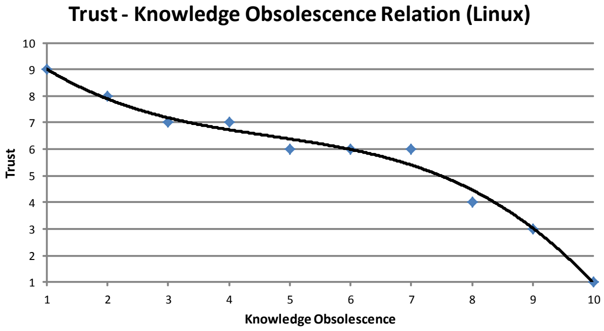

Fig. 3. Trust and Knowledge Obsolescence Relation for Linux - Final study

Selection of third order curves was guided by the fact that, in both cases, the curves were associated to the best values of the SSE and R^2 and also provided the best visual fitting.

2) Preliminary conclusions: After analyzing the results from this confirmation study, the following preliminary conclusions were drawn:

- There is an inverse relation between trust and knowledge obsolescence, that is, when knowledge obsolescence augments, the trust in the person is diminished
- Since the curves are of third order, the rate of change of trust cannot be interpreted as constant, that is, Knowledge obsolescence appears to have more influence on trust in the extremes
  - When knowledge obsolescence is low (rating between 1 and 4) or high (rating between 7 and 10) the rate of change is greater than in the middle (rating between 4 and 7)
- No apparent difference was found in the relation of trust and knowledge obsolescence between the two topics

We considered the second conclusion as a very important one. Obsolescence of knowledge appears to have more influence on trust in the extremes. That is, when obsolescence of knowledge is low (rating between 1 and 4) or high (rating between 7 and 10) the rate of change is greater than in the middle (rating between 4 and 7). Therefore, when rating trust between parties, the obsolescence of knowledge has a major impact in the extremes.

V. Implementation: Simulating Trust and Knowledge Obsolescence Relations

An important part of this thesis is devoted to the implementation of a software simulator[5]. This section describes the principal design decisions and the architecture of the software.

The main goal of the simulator is to simulate a trust network with users, items and relationships between users, and between users and items. Generation of synthetic data has been recognized as a valid technique for empirical validation in several research areas [120][121], including software engineering [122][123]. However, as stated in [124], care must be taken when conducting simulation based studies regarding validity of both data and models. This is not easy in the field of trust-based recommender systems as it is a relatively new research field in which there are not much *in vivo* or *in vitro* experiments to analyze and to compare with possible *in virtuo* or *in silico*. Therefore, and although many of the trust-based systems experiments are *in virtuo* of *in silico*, we discuss some threats to validity:

- Not much work has been carried out in the trust-based recommendations research area. Thus, we lack a solid *real* foundation to design simulation models.
- At this stage of development, we could say that most of the experimentation and results of this and other works is more algorithmically than social oriented. This means that most of the results can be taken into account by other researchers to improve algorithms with no major problems, but care must taken when working with these results in the social aspects of recommender systems.

According to the types of simulations defined by Law and Kelton [125], the type used here is *terminating simulation* because there exist a *natural* event E that specifies the length of each run. The event E in this work is a function of the number of nodes and the number of relationships between nodes. As expected, these parameters are set at the beginning of the simulation, a distinctive characteristic of *terminating simulations* [125].

Although not exempt of some debate and challenges [126], releasing software artifacts is becoming increasingly important when reporting research results [118]. Therefore, this section also provides some software artifacts (especially design related) to describe the software and its functionality.

Regarding to this domain model, we can observe the following aspects (the associations in the domain model are written following the general reading-conventions up-to-down and left-to-right):

A. Simulating Users, Items and Relationships

The main data structure used in the simulator is a *property graph*. A *property graph* is essentially a graph $G = (V, E)$ where V is the set of vertices (users, items) and E is the multiset[6] of edges formed by pairs of vertices (trust and rating relationships), but with the possibility of adding multiple *key-value* pairs (or attributes) on each of the elements of the graph[7] [134].

Before continuing in this section, it is important to note that, although nodes, users and items could mean the same in some contexts, there is an important difference between them: node is the element created in the database while users and items are human concepts (which are finally created as nodes in the database).

[5] Available at http://www.github.com/pcruzn/.

[6] Concept of multiset in mathematics is a generalization of the concept of set in which elements are allowed to appear more than once

[7] http://neo4j.com/developer/graph-database/, reviewed at 24-02-2015

The simulator works with four kinds of nodes:

- Start node: represents the source node. That is, the user getting the recommendation.
- In Between nodes: represent the nodes that are in between the start node and the end-line nodes.
- End-line nodes: represent the nodes that are in direct relationship with items. That is, these are the nodes that rate items.
- Item nodes: represent the abstract items that are recommended.

In addition to the nodes, two relationships are recognized:

- Trust relationship: a relationship of type $TRUSTS$ with possible values between 1 and 10.
- Rating relationship: a relationship of type $RATING$ with possible values between 1 and 5.

Given these definitions, the software simulates:

- The creation of one start node, several in-between nodes, several end-line nodes and several item nodes.
- The creation of trust relationships between:
  - Start node and in-between nodes
  - Start node and end-line nodes
  - In-between nodes
  - In-between nodes and end-line nodes
- The creation of rating relationships between:
  - End-line nodes and item nodes

It is possible for the software to create cycles in the graph. But, as explained in section V-C, cycles are not considered when making recommendations. It is not possible, however, for the software to create a $RATING$ relationship between the $Start$ node and any item because it will not be necessary a recommendation of an item that is already *known* by the Start node. It is also not possible for the simulator to create a trust relationship from in-between nodes or end-line nodes to the start node. This is not considered because it has no sense in the simulation of recommendations.

B. Neo4j and Property Graphs

Neo4j is a NoSQL graph database implemented in JAVA and Scala [133][134]. It is the database used in the simulator to create and manage all the nodes and relationships. *Labels* are used in Neo4j to group nodes with similar characteristics. In this simulator, four labels are used:

- Start: there is only one node labeled as Start (this is the simulated user that gets the recommendation).
- InBetween: all nodes between the Start node and the final nodes (the ones that rate items) are labeled with the InBetween label.
- EndLine: the final nodes (i.e., the nodes that rate the items) are labeled as EndLine.
- Item: all items are labeled as Item.

Figure 5 depicts an example graph with colors for the start node (blue), in-between nodes (green), end-line nodes (light blue) and item nodes (orange). The figure also shows the relationships of type TRUSTS and RATING. The nid shown in each node is the attribute (property key) used for identification of each node with a particular value (property value). The $nids$ are shown in the figure, but they are of interest only to the simulator.

C. Getting Trust On a User

Computing trust on a user (a sink node in the database) is done by an algorithm implemented as a *Transcaction Script* in a class *UserService*. The service requires two nodes: the source node and the sink node. Although the implementation allows computing the trust between any source-sink node pairs, for practical reasons the service is only used for the start node and an end-line node. The algorithm developed for this simulator consists in getting all simple paths (i.e., no cycles in the graph) and, for each path, taking the average of the trust relationships between nodes. Finally, the average of the trust for each path is computed and returned as the trust on the user. This is depicted in algorithm 1.

```
Input: sourceNode, sinkNode
Output: trustOnUser
paths = getAllSimplePaths(OUTGOING, sourceNode, sinkNode)
while paths.hasNext() do
    relationships = getAllRellationshipsInPath
    while relationships.hasNext() do
        store relationship value (trust)
    end while
    compute average of trust for all relationships in a path
    store average trust in a path
end while
trustOnUser = compute average of trust of all paths
return $trustOnUser$
```

Algorithm 1: Algorithm for getting trust on a user

D. Design and Architecture

Software design of the simulator is mainly influenced by some commonly known design patterns, especially the *Abstract Factory* pattern [131] and other enterprise architecture design patterns, especially the *Transaction Script* and *Domain Model* patterns [132].

While *Abstract Factory* pattern was used to decouple creation of nodes, *Transaction Script* and *Domain Model* pattern were used to organize all interaction logic between the client (in this case, the Java application) and the server (in this case, the embedded database).

The main services created by using *Transaction Script* pattern are (see figure 4):

- GraphService
- ItemService
- RecommenderService
- RelationshipService
- UserService

E. General Description of the Simulator's Components Interaction

Figure 4 shows a general description of the messages sequence between the main simulator's components. This diagram was left to the end of this section as it requires some knowledge of all of the components. However, the reader should note this diagram is a useful complement to section V-A.

The diagram shows that all simulations start with an indication of the user to start the simulator. Once simulator is started, it begins to create user nodes, item nodes, relationships between all nodes (following the specifications given before in this section) and finally requesting a recommendation. The diagram also shows that the main controller process records the elapsed time for each simulation.

VI. Validation of the Simulator's Components

Before running simulations scenarios, the simulator's components require validation. That is, we first need to delve attention to the outputs of the main simulator's components (functional validation).

In this work, the simulator's engine has two main responsibilities:

- Simulating the values for ratings (between users and items). This part of the simulator's engine generates two kinds of ratings: good-item ratings and random ratings (this is described in previous sections).
- Simulating the values of trust and knowledge obsolescence (between users).

It should be noted that bigger networks are impossible to show in a document. Therefore, this section implicitly shows validation for:

- Creation of nodes (Start node, InBetween nodes, EndLine nodes)
- Creation of relationships (trust and knowledge obsolescence, and ratings)
- Getting the values of ratings for items
- Getting a recommendation of items

Section VI-A presents validation for the simulation engine.

A. Simulation Engine

In this work, the simulation engine concept is used to denote the ratings and trust generators. The simulation engine consists of two big components:

- Rating Generator: this component generates ratings for good-quality and random(neutral or uniform)-quality items.
- Trust Generator: this component generates the trust values for relationships.

In general, all the generators from the engine are implemented using the *inverse-transform* method (see [125] for further information). Previous experience was used to devise the theoretical populations in both generators [2][27].

Validation of both generators was carried on by using the *chi-square* goodness of fit test. This test requires a simulated random sample of size n from the population we are trying to achieve [128]. Every statistical testing was performed with a random sample of size $n = 4000$ (i.e., 4000 ratings or trust values generated)[8] and by approximating the hypothetical distribution using the expected relative frequencies and the sample size.

B. Validation of a Recommendation

To validate a recommendation, we run a small simulation (see 5). First step in obtaining a recommendation is to get, for each item, the raters set which will be denoted here by R_i, where i is the item id. In this case:

- For item $it1$, $R_{it1} = \{e1, e2, e3\}$
- For item $it2$, $R_{it2} = \emptyset$
- For item $it3$, $R_{it3} = \{e2, e3\}$

The next step is to determine the trust on the user by using the algorithm exposed in section V. By applying manually the algorithm, we get the trust for the users (denoted here by $T(n)$, where n is the node) from the source node ($i1$).

From the source node to the sink node $e1$ there are four valid paths:

- $i1, e1$
- $i1, n2, n1, e1$
- $i1, n3, n1, e1$
- $i1, n3, e1$

Thus, we can compute the trust on $e1$ as:

$$T(e1) = \frac{6 + \frac{1+8+10}{3} + \frac{3+6+10}{3} + \frac{3+6}{2}}{4} \approx 5.8$$

From the source node to the sink node $e2$ there are two valid paths:

- $i1, n2, e2$
- $i1, n3, n1, n2, e2$

In this case, trust on $e2$ is computed as:

$$T(e2) = \frac{\frac{1+8}{2} + \frac{3+6+9+8}{4}}{2} = 5.5$$

The reader should note that, as an example, a path like $i1, n2, n1, n2, e2$ is not a valid one because node $n2$ appears two times (i.e., there is a cycle in the path).

Finally, trust for node $e3$ is computed:

$$T(e3) = \frac{1 + \frac{1+3}{2} + \frac{1+8+3}{3} + \frac{3+6+3}{3} + \frac{3+6+9+3}{4}}{5} = 3.25$$

considering the following valid paths:

- $i1, e3$
- $i1, n2, e3$
- $i1, n2, n1, e3$
- $i1, n3, n1, e3$
- $i1, n3, n1, n2, e3$

[8] Chi-square test is very sensitive to sample size. However, as we are generating *in silico* these random numbers, we can use a value of $n = 4000$ for the sample size, which is higher than the commonly cited *ideal* value of $n = 1000$ [129][130].

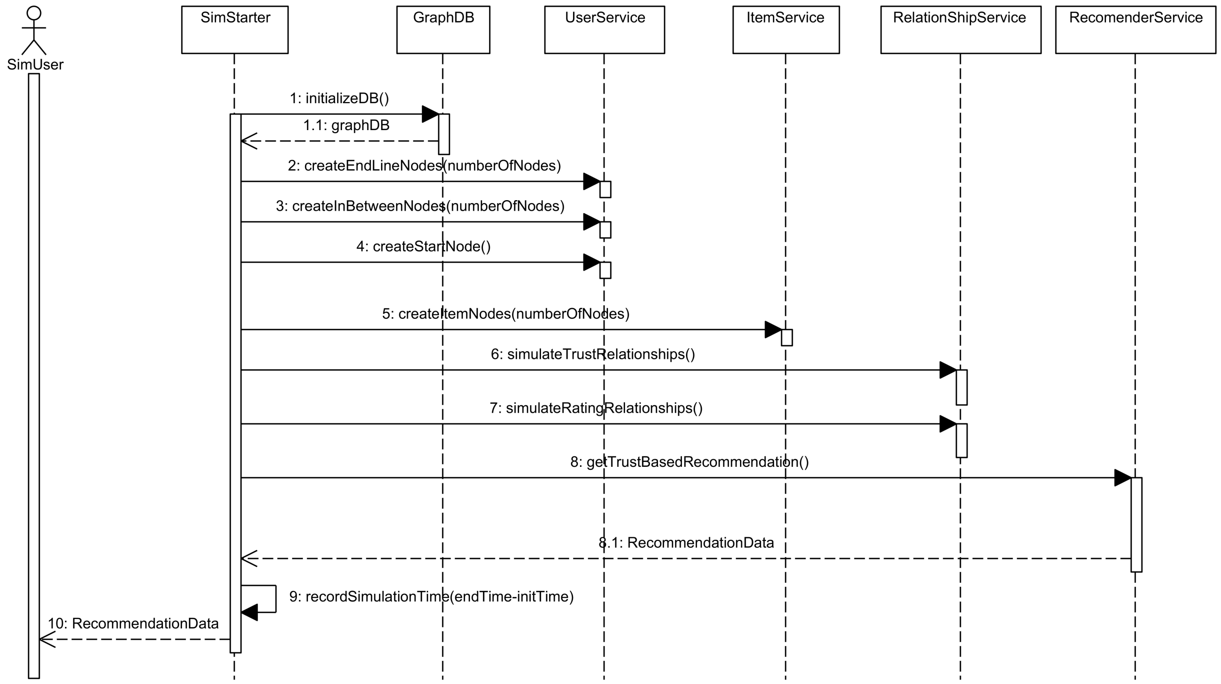
SimUser
SimStarter
GraphDB
UserService
ItemService
RelationShipService
RecomenderService
1: initializeDB()
1.1: graphDB
2: createEndLineNodes(numberOfNodes)
3: createInBetweenNodes(numberOfNodes)
4: createStartNode()
5: createItemNodes(numberOfNodes)
6: simulateTrustRelationships()
7: simulateRatingRelationships()
8: getTrustBasedRecommendation()
8.1: RecommendationData
9: recordSimulationTime(endTime-initTime)
10: RecommendationData

Fig. 4. Sequence diagram for components of the simulator.

In this example, we instructed the simulator to consider only users with $T(n) \geq 5.0$ for making the recommendation. Therefore, the ratings from users $e1$ and $e2$ are considered.

For item 1, the rating is simply the average of the ratings from $e1$ and $e2$:

$$r(it1) = \frac{1+5}{2} = 3$$

For item 3, the rating is 5, i.e., the rating from $e2$ ($e3$ is not considered as the trust from the source node is not enough).

All values shown here were compared to the output of the simulator. Similar to this example, other seven networks were simulated and the same procedure applied. This procedure helped to check the functional correctness of the software by observing its outputs.

VII. Simulation and Results

Several simulation runs were used to generate data in order to reach some conclusions. As there are several parameters to be set in the simulator, we defined to big groups of simulations:

1) Trust-only group: networks with trust relationships only.
2) Trust-knowledge-obsolescence group: networks with both trust and knowledge obsolescence relationships.

Both groups consist of five cases. Each case represent a particular parameter setting variation. In the first group, the parameter that was varied along runs was the *trust relationship ratio* with values $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ (five cases). In the second group, trust relationship ratio was varied as in the first group, and the probability for a knowledge obsolescence relationship was set to 0.3. Thus, for the second group we have the following five cases: $\{\{0.1\}\{0.3\}, \{0.2\}\{0.3\}, \{0.3\}\{0.4\}, \{0.4\}\{0.3\}, \{0.5\}\{0.3\}\}$.

Each case within a group consists of 40 simulation runs. This gives a total of 400 simulation runs (40 runs for each of the 5 cases within 2 groups). Much discussion has been devoted to the length of simulations (i.e., how many simulations should be run in order to generate statistical-interesting data) [135][136][137][138][139]. As noted in [135], in stochastic simulation it is not easy to determine how many runs as in the case of deterministic simulation (in this case, only one run is required). Although we are working with *in silico* models, the reader should remember we are actually simulating human behavior. Thus, in this work we defined a total of 40 simulation runs for each case as this number is commonly used in qualitative analysis [140][141][142].

The following other settable parameters were fixed for both groups[9]:

- Start node: by definition, only one start node can exist in the simulation.
- InBetween nodes: 20
- EndLine nodes: 10
- Item nodes: 50
- Good-item ratio: 0.8
- Rater node-Item node connectivity factor: 0.3

Therefore, a maximum of 6480 relationships could exist[10] (including rating relations) in each one of the 400 networks

[9] As no previous related studies exist, we fixed these parameters considering execution time as an important factor. Higher values on any of these parameters result in very large simulation times (more than 2 hour for each case).

[10] A network is treated as a directed graph.

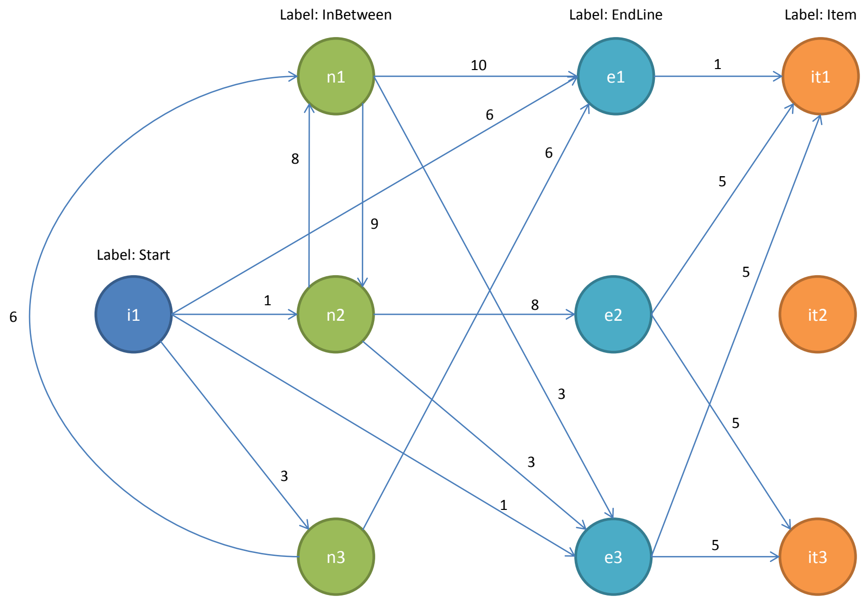

Fig. 5. Network example for validation.

(cases) generated.

Simulating this kind of networks is a time and space consuming process. However, the time and space consuming characteristic is an inherent characteristic of almost every complex simulation process. Therefore, the reader should not confuse the time and space consuming aspect of simulating a network (relationships and recommendations) with *working* with a real network. It is expected that working with a real network be a less time and space consuming process as the relationships generation process occurs in real-time. Probably, most time and space consuming process in a real network is the recommendation process.

First group of simulation runs took an actual time of 51 minutes while the second one took 52 minutes. Therefore, we have a total of 1.7 hours of actual simulation time [11].

As a matter of clarification for the following sections, some conventions will be used:

- Each case within a group is numbered from 1 to 5.
- Each test case (referring to hypotheses testings, not to software testing) is numbered from C1 to C5. These cases represents parallel comparisons between the two groups. For example, C1 represents a comparison between trust-only approach with trust relationships ratio of 0.1 and a trust-knowledge-obsolescence approach with trust relationships ratio of 0.1 and knowledge obsolescence probability of 0.3.

A. *Success Scenarios*

Although not common in the literature as we would want, a ratio of success scenarios is an important metric. We define a **success scenario** as *a network of nodes (users) with trust and/or knowledge obsolescence relationships in which at least one item can be recommended*. This definition needs to be complemented with the two possible cases in which **items cannot be recommended**. These cases are:

- There are no nodes rating items (very unlikely in this work, as the probability for a rater node to rate an item was set to 0.3).
- The algorithm could not find a way to reach a rater node (most likely).

Now it should be clear why the ratio of success scenarios is of high importance: it is indicating success of a recommender system. The lower the ratio of success scenarios, the higher the ratio of recommender system failure.

As stated before in the first part of this section, for each case of parameter setting in the simulator we run 40 simulations. Therefore, the ratio (or percentage) is always measured over 40.

Table I shows the trust relationships ratio (parameter for the simulation), the number of success scenarios and the ratio. The table also shows a number to identify each case (cases from 1 to 5).

In a similar way, table II shows the trust relationships ratio, the knowledge obsolescence probability, the number of success scenarios and the ratio. In this case, the trust relationships ratios and the knowledge obsolescence probability are both

[11] Simulation time includes *actual time* between starting of the simulator and the simulation run ending. It does not include any other time.

Case	Trust Rels. Ratio	Success Scenarios	Ratio
1	0.1	28	0.70
2	0.2	35	0.88
3	0.3	38	0.95
4	0.4	39	0.98
5	0.5	40	1.00

TABLE I
NUMBER AND RATIO OF SUCCESS SCENARIOS (TRUST ONLY).

parameters for the simulation. Again, a case number is assigned (from 1 to 5).

Case	Trust Rels. Ratio	Success Scenarios	Ratio
1	0.1	40	1.00
2	0.2	39	0.98
3	0.3	40	1.00
4	0.4	40	1.00
5	0.5	40	1.00

TABLE II
NUMBER AND RATIO OF SUCCESS SCENARIOS (TRUST W/KO PROBABILITY = 0.3).

Figure 6 shows a plot of the evolution of both ratios for each case. In the plot, we can observe the *asymptotic aspect* of the ratio when only trust relationships are used.

Observing data from tables I and II and plot in figure 6 we can state the following preliminary conclusions:

- First, in the case of networks with trust relationships only, it is interesting to note how the ratio of success scenarios quickly increases as the trust relationships ratio increases (asymptotic behavior).
- Second, in the case of networks with trust and knowledge obsolescence relationships, all five cases shown a ratio near to 40.
- Third, it seems knowledge obsolescence is a determinant factor on the recommender system success in *less trustworthy*[12] scenarios.

B. Coverage of Items

A well-known metric to measure performance of recommender systems (whether trust-based or not) is called Coverage. In its simplest form, coverage is the ratio of the number of recommended items over the total number of items. Thus, coverage can be defined as follows:

$$Coverage = \frac{NumberOfRecommendedItems}{TotalItems}$$

Table III shows the average number of recommended items and the standard deviation for each case when using the trust-only approach. Table IV shows the average number of recommended items and the standard deviation for each case when using trust and knowledge obsolescence. Figure 7 shows a plot with a visual comparison of the average number of recommended items for both approaches.

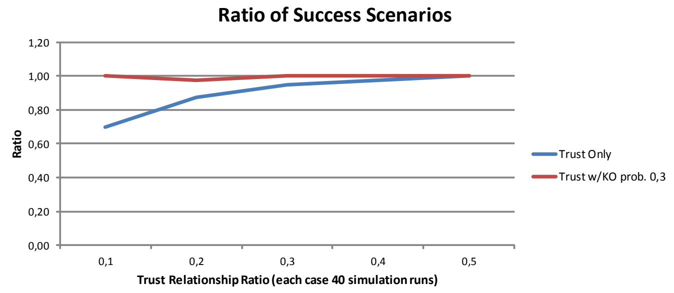

Fig. 6. Comparison of the ratios of success scenarios for each case.

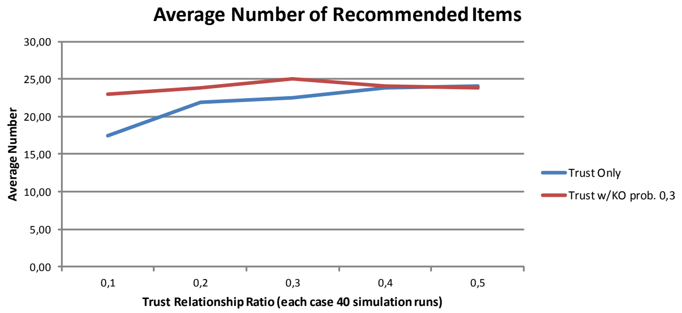

Fig. 7. Average Number of Recommended Items.

Case	Trust Rels. Ratio	Avg. of Items	Std. Dev.
1	0.1	17,46	7,47
2	0.2	21,86	7,82
3	0.3	22,55	5,69
4	0.4	23,82	4,28
5	0.5	24,05	4,94

TABLE III
AVERAGE NUMBER OF RECOMMENDED ITEMS (TRUST ONLY)

Case	Trust Rels. Ratio	Avg. of Items	Std. Dev.
1	0.1	23	4,17
2	0.2	23,87	6,49
3	0.3	25	3,31
4	0.4	24,05	4,14
5	0.5	23,88	2,59

TABLE IV
AVERAGE NUMBER OF RECOMMENDED ITEMS (TRUST W/KO PROBABILITY = 0.3)

[12] A *trustworthy scenario* is one in which trust has been widely used as relationships between nodes (i.e., trust has been widely used by users).

As we have fixed the total number of items to 50, we can calculate the coverage for each case. We can make use of a statistical approach in order to test whether the proportions of recommended items (for both approaches) are different or not. Therefore, our interest is in testing the following hypotheses:

$$H_0 : p_1 = p_2$$
$$H_1 : p_1 \neq p_2$$

Our null hypothesis state that proportions (coverages) are equal for both approaches. The alternative hypothesis states that the coverage given by a trust-only-based recommender system is less than the coverage given by a trust-knowledge-obsolescence-based recommender system. In the hypotheses formulation, p_1 represents the coverage for *trust-only* cases and p_2 represents the coverage for *trust with knowledge obsolescence* cases.

To test these hypotheses, we use the following test statistic:

$$Z_0 = \frac{\hat{P}_1 - \hat{P}_2}{\hat{P}(1-\hat{P})(\frac{1}{n_1} + \frac{1}{n_2})}$$

where n_1 and n_2 are the sample sizes. We have available an estimator for $\hat{P}$ defined as:

$$\hat{P} = \frac{X_1 + X_2}{n_1 + n_2}$$

Rejection criteria for the null hypothesis is $z_0 < -z_{\alpha/2}$. With a significance level of $\alpha = 0.05$ (confidence level of 0.95), we have $z_{0.025} = 1.96$. Thus, we will reject the null hypothesis if $z_0 < -1.96$.

Test Case	$\hat{p}_1$	$\hat{p}_2$	z_0	H_0
C1	0.56	0.8	-2.57	Rejected
C2	0.7	0.78	-0.91	Not rejected
C3	0.76	0.8	-0.48	Not rejected
C4	0.78	0.8	-0.25	Not rejected
C5	0.8	0.8	0	Not rejected

TABLE V
INFERENCE FOR THE DIFFERENCE IN PROPORTIONS (COVERAGE).

According to table V, we have statistical evidence to state that coverage in the trust-knowledge-obsolescence approach is greater than in the trust-only approach for test case C1 (i.e., a ratio of 0.1 for trust relationships and a 0.3 for knowledge obsolescence relationship probability). For the other test cases, although the proportions were always higher for the trust-knowledge-obsolescence approach (except in C5), we lack strong statistical support for stating that coverages are different.

C. Quality of Recommendations

Quality is hard to define in any area. In this work, we used the Goal-Question-Metric (GQM) approach to give some insight on what quality of recommendations means in this context and to define a metric to measure this quality.

Using GQM we state the following elements:

- Goal: (I as a user) Want to get the best items (i.e., highest rated items)
- Question: Does this approach (simulation) give (me) the best items (i.e., highest rated items)?
- Metric: Average rating of recommended items per case of simulations

It should be remembered here that we generated rating relationships in a controlled manner (i.e., not uniform or *random* approach), thus allowing to get measurements for this metric easily.

Table VI shows the average of ratings of the recommended items when only trust is used. On the other hand, table VII shows the average of ratings of the recommended items when trust and knowledge obsolescence are used.

Case	Trust Rels. Ratio	Avg. Rating	Std. Dev.
1	0.1	4,25	0,42
2	0.2	4,18	0,31
3	0.3	4,14	0,22
4	0.4	4,13	0,21
5	0.5	4,16	0,21

TABLE VI
AVERAGE OF RATINGS OF RECOMMENDED ITEMS (TRUST ONLY).

Case	Trust Rels. Ratio	Avg. Rating	Std. Dev.
1	0.1	4,17	0,25
2	0.2	6,49	0,26
3	0.3	3,31	0,21
4	0.4	4,14	0,25
5	0.5	2,59	0,22

TABLE VII
AVERAGE OF RATINGS OF RECOMMENDED ITEMS (TRUST W/KO PROBABILITY = 0.3).

Figure 8 presents a plot of the average of the ratings of the items that appeared as recommended. Neither visually nor numerically we find significant differences in the ratings of the recommended items using trust only and trust with knowledge obsolescence. Given this insight, we use again a statistical approach to confirm our perceptions. In this case, we want to test the following hypotheses:

$$H_0 : \mu_1 - \mu_2 = 0$$
$$H_1 : \mu_1 - \mu_2 \neq 0$$

What these set of hypotheses are stating is that the average of ratings for the trust-only approach (μ_1) is equal to the average of ratings for the trust-knowledge obsolescence approach (μ_2). Thus, the hypotheses could be easily rewritten as follows:

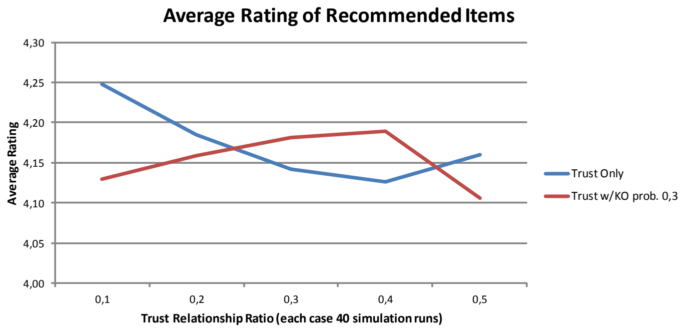

Fig. 8. Average of ratings of the recommended items.

$$H_0 : \mu_1 = \mu_2$$
$$H_1 : \mu_1 \neq \mu_2$$

which is a bit clear for our purposes (we want to test if there are significant differences between average ratings of recommended items by the two approaches).

As the values of the variances for the average of ratings are similar, we argue the population variances are equal. Also, the reader should remember all of the values for ratings were simulated in a similar way (i.e., using same procedure) and there is no element in the simulator that could interfere or bias any of the two variances. Thus, we can argue homoscedasticity in the random variables. However, we cannot argue that variances are known in advance. We have no previous experience with this simulator (as this was developed for this work thesis), thus we cannot state that variances are known in advance, although we have the sample variances. Therefore, we will make use of a $t - student$ distribution in order to evaluate the hypotheses.

Our test statistic is defined as follows:

$$T_0 = \frac{\bar{X}_1 - \bar{X}_2}{S_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

T_0 follows a $t-student$ distribution with n_1+n_2-2 degrees of freedom. S_p is the square root of a pooled estimator of σ^2:

$$S_p^2 = \frac{(n_1 - 1)S_1^2 + (n_2 - 1)S_2^2}{n_1 + n_2 - 2}$$

Rejection criteria for the null hypothesis are $t_0 > t_{\alpha/2,n_1+n_2-2}$ or $t_0 < -t_{\alpha/2,n_1+n_2-2}$.

With a significance level of $\alpha = 0.05$ (confidence level of 0.95), we construct table VIII which shows the results for the tests for each of the five comparisons (we use the same naming convention for the comparisons; see first part of this section for more on this).

For all cases, we make use of $t_{0.0025,n_1+n_2-2} = 2$. Values for n_1 and n_2 are not shown in table VIII as they were already shown in tables VI and VII.

Test Case	$\bar{x}_1$	$\bar{x}_2$	s_1^2	s_2^2	t_0	H_0
C1	4.25	4.13	0.17	0.06	1.50	Not rejected
C2	4.18	4.16	0.09	0.07	0.30	Not rejected
C3	4.14	4.18	0.05	0.05	-0.80	Not rejected
C4	4.13	4.19	0.04	0.06	-1.08	Not rejected
C5	4.16	4.11	0.04	0.05	1.01	Not rejected

TABLE VIII
INFERENCE FOR THE DIFFERENCE IN AVERAGES OF THE RATINGS.

According to table VIII, we do not have strong statistical evidence to state that the average of the ratings of items recommended with a trust-only approach is different to the average of the ratings of items recommended with a trust-knowledge-obsolescence approach in any of the five comparisons (test cases).

VIII. CONCLUSIONS, CHALLENGES AND FUTURE WORK

A. *Conclusions*

This work led us to interesting conclusions about the potential use of knowledge obsolescence in recommender systems.

First, we found an **interesting behavior in the number of *success scenarios*[13] when knowledge obsolescence is used**. When only trust relationships are used, an asymptotic behavior of the curve given by the number of the success scenarios as trust relationship ratio increases was found. When using knowledge obsolescence, the curve was much more stable, reaching (except in one case) the 100% of success. This difference in ratios given by the different approaches led us to the following interpretation: **in scenarios in which a trust relationship is hard to define, knowledge obsolescence could have a significant impact in the success of a recommender system**. It should be noted here that a success scenario is a case of recommender system success (i.e., a failed scenario is also a case in which a recommender system would fail).

Second, except in one case, we found trust-knowledge-obsolescence scenarios giving more items recommended (higher coverage). However, statistical evidence strongly supports the case in which there are a 10% of trust relationships and the probability for knowledge obsolescence relationships is 0.3. This led us to an interesting general conclusion and interpretation: **knowledge obsolescence relationships would improve items coverage when used in areas subject to knowledge obsolescence**. Thus, in contexts where knowledge obsolescence does not have importance (i.e., users would not be able to rate knowledge obsolescence on partners) will not have measurable effects on items coverage. This interpretation is founded on the fact that in areas in which knowledge obsolescence is not important, the probability of knowledge obsolescence relationships will be low (thus, the network would become similar to a trust-only one).

Third, **we found no significant effect of knowledge obsolescence neither numerically nor statistically on the**

[13] A success scenario is defined as a network in which at least one item can be recommended.

quality of the recommendation. That is, we were not able to find higher-rated items by using trust and knowledge obsolescence together (compared to the case in which only trust relationships are used).

B. Challenges: Threats to Validity and Implementation

1) Threats to Validity: As noted in previous sections, threats to validity always exist when working with *in silico* models. In our case, the main threats to validity are:

- We treated *items* as *generic returnable items* by a Web site. These generic returnable items can be movies, books, papers, among others. However, attention should be put when extending our results to specific returnable items. For example, these results could apply to movies, but not necessarily to books.
- The algorithm used for recommendations should be considered too. We implemented a specific algorithm (see section V) and the results are specific to this algorithm. Extension of our results to other recommendation systems should be done carefully.
- Implicitly we assumed *human behavior*. Many of these assumptions were given by observation to several *virtual communities*. Extension of our results to specific virtual communities should consider previous knowledge on user behavior.

2) Implementation: One interesting challenge we faced in the development of this thesis work was considering appropriate principles from Software Engineering for the development of the software simulator. Main efforts were devoted to software design and the overall architecture. The efforts on the software design and on the overall software architecture were motivated by the requirement of developing a scalable software considering possible improvements and new parameter settings in the future. This will allow us to continue several other types of simulations to study other aspects of knowledge obsolescence inclusion in trust-based recommender systems.

Another challenge we faced is related to the execution of the software simulator. As stated before, simulations are often time and space consuming processes. In our specific context, the simulator developed was specifically space consuming. We found several execution interruptions by exceptions related to the memory assigned to the Java Garbage Collector (this is a common performance issue, as noted by Trisha Gee in the chapter “Why Java 8?” from [143]). What we did was to re-set some related parameters when starting the Java Runtime Environment. Fortunately, this is an issue that has been reported several times by other software products and servers. Thus, the solution is relatively known.

C. Future Work

As a future work, we are interested in exploring the effect of the size of the networks simulated in the trust-knowledge-obsolescence approaches. As designed, the software simulator allows us to control the size of groups of nodes in the network. For example, we can freely vary the size of the InBetween group of nodes in order to see if there is some effect on the recommendations. Also, we would like to explore in more detail the effects of varying the Rater-node-Item-node connectivity factor on the coverage and quality of the recommendations. As a final remark regarding future work, we are also interested in exploring information retrieval techniques to build the knowledge obsolescence indicator in a semi-automatic way.

REFERENCES

[1] W. M. Evan, *The Problem of Obsolescence of Knowledge*, IEEE Transactions on Engineering Management 10(1) (pp. 29-31), 1963.

[2] P. Cruz, H. Astudillo, *Exploring the Trust and Obsolescence of Knowledge Relation*, 32nd International Conference of The Chilean Computer Science Society, Temuco - Chile, 2013.

[3] W. Sherchan, S. Nepal, C. Paris, *A Survey of Trust in Social Networks*, ACM Comput. Surv. 45(4), 2013.

[4] J. Barnes, *Class and Committees in a Norwegian Island Parish*, Human Relations 7(1) (pp. 39-58), 1954.

[5] B.D. Adams, M. H. Thomson, A. Brown, J. A. Sartori, T. Taylor, S. Waldherr, *Organizational Trust in the Canadian Forces*, National Defence and the Canadian Forces, 2008.

[6] B.D. Adams, R.D.G. Webb*Trust in Small Military Teams*, Proceedings of the 7th International Command and Control Research and Technology Symposium, 2002.

[7] F. Tonkiss, *Trust, Confidence and Economic Crisis*, Intereconomics, Vol. 44 Issue 4 (pp. 196-202), 2009.

[8] R. Borum, *The Science of Interpersonal Trust*, The MIitre Corporation, 2010.

[9] B. J. Starnes, S. A. Truhon, V. McCarthy, *A Primer on Organizational Trust*, American Society for Quality, 2010.

[10] W. M. Rahn, J. E. Transue, *Social Trust and Value Change: The Decline of Social Capital in American Youth, 1976-1995*, Political Psychology, Vol. 19 Issue 3 (Special Issue: Psychological Approaches to Social Capital) (pp. 545-565), 1998.

[11] C-M. Chiua, M-H. Hsub, E.T.G. Wangc, *Understanding Knowledge Sharing in Virtual Communities: An Integration of Social Capital and Social Cognitive Theories*, Decision Support Systems, Vol. 42 Issue 3 (pp. 1872-1888), 2006.

[12] A.A. Adamopoulou, A.L. Symeonidis, *A Simulation Testbed for Analyzing Trust and Reputation Mechanisms in Unreliable Online Markets*, Electronic Commerce Research and Applications, Vol. 13 Issue 5 (pp. 368-386), Elsevier, 2014.

[13] B. zcan, C. Bjrnskov, *Social Trust and Human Development*, The Journal of Socio-Economics, Vol. 40 Issue 6 (pp. 753-762), 2011.

[14] N. Luhmann, *Familiarity, Confidence, Trust: Problems and Alternatives*, Trust: Making and Breaking Cooperative Relations, Department of Sociology, University of Oxford, 2000.

[15] M. Deutsch, *Trust and Suspicion*, The Journal of Conflict Resolution, Vol. 2 Issue 4 (pp. 265-279), 1958.

[16] J. Park, H. Lee and C. Kim, *Corporate Social Responsibilities, Consumer Trust and Corporate Reputation: South Korean Consumers' Perspectives*, Journal of Business Research, Vol. 67 Issue 3 (pp. 295 - 302), Elsevier, 2014.

[17] R.A. Thacker, *The Application of Social Exchange to Commitment Bonds of Pro-union Employees: Cognitive Calculations of Reciprocity*, Human Resource Management Review, Elsevier, 2015.

[18] M. J. Hetherington, *The Political Relevance of Political Trust*, The American Political Science Review, Vol. 92 Issue 4 (pp. 791-808), 1998).

[19] W. Mishler, R. Rose, *What Are the Origins of Political Trust?: Testing Institutional and Cultural Theories in Post-communist Societies*, Comparative Political Studies February, Vol. 34 Issue 1 (pp. 30-62), 2001.

[20] B.M. Muir, *Trust in Automation: Part I. Theoretical Issues in the Study of Trust and Human Intervention in Automated Systems*, Ergonomics, Vol. 37 Issue 11 (pp. 1905-1922), 1994.

[21] J.D. Lee, K.A. See, *Trust in Automation: Designing for Appropriate Reliance*, Human Factors: The Journal of the Human Factors and Ergonomics Society, Vol. 46 Issue 1 (pp. 50-80), 2004.

[22] R.R. Hoffman, M. Johnson, J.M. Bradshaw, A. Underbrink, *Trust in Automation*, IEEE Intelligent Systems, Vol. 28 Issue 1 (pp. 84 - 88), 2013.

[23] A. Jsang, R. Ismail, C. Boyd, *A Survey of Trust and Reputation Systems for Online Service Provision*, Decision Support Systems, Vol. 43 Issue 2 (pp. 618-644), 2007.

[24] M. Vinkovits, A. Zimmermann ,*Defining a Trust Framework Design Process*, Lecture Notes in Computer Science, Vol. 8058 (pp. 37-47), 2013.

[25] M. Kinateder, E. Baschny, K. Rothermel, *Towards a Generic Trust Model Comparison of Various Trust Update Algorithms*, Lecture Notes in Computer Science, Vol. 3477 (pp. 177-192), 2005.

[26] F. Gómez, G. Martínez, *Towards Pre-standardization of Trust and Reputation Models for Distributed and Heterogeneous Systems*, Computer Standards & Interfaces, Vol. 32 Issue 4 (pp. 185-196), Elsevier, 2010.

[27] P. Cruz, O. Cornejo, H. Astudillo, *Trust-based Improved Recommendation of IT-related Web Resources*, XL Latin American Computing Conference (CLEI), 2014.

[28] R.M. Emerson, *Social Exchange Theory*, Annual Review of Sociology, Vol. 2 (pp. 335-362), 1976

[29] J. Jacobs, *The Death and Life of Great American Cities*, Random House, 1961.

[30] R. C. Mayer, J. H. Davis, F. D. Schoorman, *An Integrative Model of Organizational Trust*, The Academy of Management Review, Vol. 20 Issue 3 (pp. 709-734), 1995.

[31] S. Milgram, *The Small-World Problem*, Psychology Today, Vol. 1 Issue 1 (pp. 60-67), 1967.

[32] J. Travers and S. Milgram, *An Experimental Study of the Small World Problem*, Sociometry, Vol. 32 Issue 4 (pp. 425-443), American Sociological Association, 1969.

[33] L. Backstrom, P. Boldi, M. Rosa, J. Ugander, S. Vigna,*Four Degrees of Separation*, Proceedings of the 4th Annual ACM Web Science Conference (pp. 33-42), ACM, 2012.

[34] P. Boldi, S. Vigna, *Four Degrees of Separation, Really*, Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining, ACM, 2012.

[35] D. Easley, J. Kleinberg, *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*, Cambridge University Press, 2010.

[36] M.S. Granovetter, *The Strength of Weak Ties*, The American Journal of Sociology, Vol. 78 Issue 6 (pp. 1360-1380), JSTOR, 1973.

[37] S. Wasserman, K. Faust, *Social Network Analysis: Methods and Applications*, Cambridge University Press, 1994.

[38] J. Golbeck, *Introduction to Computing with Social Trust*, Computing with Social Trust, Springer-Verlag, 2009.

[39] Y.A. Kim, M.A. Ahmad, *Trust, Distrust and Lack of Confidence of Users in Online Social Media-sharing Communities*, Knowledge-Based Systems, Vol. 37 (pp. 438-450), 2013.

[40] E. Agichtein, C. Castillo, D. Donato, A. Gionis, G. Mishne, *Finding High-quality Content in Social Media*, Proceedings of the 2008 International Conference on Web Search and Data Mining (pp. 183-194), ACM, 2008.

[41] G. Virgil, J.M. Wing, *Towards a Theory of Trust in Networks of Humans and Computers*, Proceedings of the 19th International Conference on Security Protocols (pp. 223-242), Springer-Verlag, 2011.

[42] T. Grandison, M. Sloman, *A Survey of Trust in Internet Applications*, IEEE Communications Surveys & Tutorials 3(4) (pp. 2-16), 2009.

[43] V. Peñarroja, V. Orengo, A. Zornoza, J. Sánchez, P. Ripoll, *How Team Feedback and Team Trust Influence Information Processing and Learning in Virtual Teams: A Moderated Mediation Model*, Computers in Human Behavior, Vol. 48 (pp. 9-16), Elsevier, 2015.

[44] B. Bhargava, L. Lilien, A. Rosenthal, M. Winslett, M. Sloman, T.S. Dillon, E. Chang, F.K. Hussain, W. Nejdl, D. Olmedilla, V. Kashyap, *The Pudding of Trust*, IEEE Intelligent Systems, Vol. 19 Issue 5 (pp. 74 - 88), 2004.

[45] Z. Weiliang, V. Varadharajan, *Trust Management for Web Services*, IEEE International Conference on Web Services (pp. 818-821), 2008.

[46] C. Safran, D.Z. Sands, D.M. Rind, *Online Medical Records: A Decade of Experience*, Methods of Information In Medicine, Vol. 38 (pp. 308-312), 1999.

[47] R.J. Anderson, *Clinical System Security: Interim Guidelines*, British Medical Journal (pp. 109111), 1996.

[48] R.J. Anderson, *A Security Policy Model for Clinical Information Systems*, IEEE Symposium on Security and Privacy (pp. 30-42), 1996.

[49] M. J. Bates, *Understanding Information Retrieval Systems: Management, Types, and Standards*, Auerbach Publications, 2011.

[50] U.G. Wilhelm , S. Staamann, L. Buttyán, *On the Problem of Trust in Mobile Agent Systems*, NDSS Symposium, 1998.

[51] M. Blaze, J. Feigenbaum, P. Resnick, M. Strau, *Managing Trust in an Information-Labeling System*, Transactions on Emerging Telecommunications Technologies 8(5) (pp. 491-501), 1997.

[52] R.N. Taylor, N. Medvidovic, E.M. Dashofy, *Software Architecture: Foundations, Theory, and Practice*, Wiley Publishing, 2009.

[53] Y. Han, S. Zhiqi, M. Chunyan, A. Bo, *Challenges and Opportunities for Trust Management in Crowdsourcing*, IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT) Vol. 2 (pp. 486-493), IEEE, 2012.

[54] H. Zhao, X. Yang, X. Li,*cTrust: Trust Management in Cyclic Mobile Ad Hoc Networks*, IEEE Transactions on Vehicular Technology 62(6) (pp. 2792-2806), IEEE, 2013.

[55] Z. Su, L. Liu, M. Li, X. Fan, Y. Zhou, *ServiceTrust: Trust Management in Service Provision Networks*, IEEE International Conference on Services Computing (SCC) (pp. 272-279), IEEE, 2013.

[56] R. Roman, P. Najera, J. Lopez, *Securing the Internet of Things*, Computer 44(9) (pp. 51 - 58), IEEE, 2011.

[57] C. Chen, S. Helal, *A Device-centric Approach to a Safer Internet of Things*, Proceedings of the 2011 International Workshop on Networking and Object Memories for the Internet of Things (pp. 1-6), ACM, 2011.

[58] W. Ren, *QoS-aware and Compromise-resilient Key Management Scheme for Heterogeneous Wireless Internet of Things*, Int. J. Netw. Manag. 21(4) (pp. 284-299), John Wiley & Sons, 2011.

[59] Z. Liang, C. Han-Chieh, *Multimedia Traffic Security Architecture for the Internet of Things*, IEEE Network 25(3) (pp. 35-40), IEEE, 2011.

[60] D. Chen, G. Chang, D. Sun, J. Li, J. Jia, X. Wang, *TRM-IoT: A Trust Management Model Based on Fuzzy Reputation for Internet of Things*, Computer Science and Information Systems 8(4) (pp. 1207-1228), 2011.

[61] F. Bao, I-R. Chen, *Trust Management for the Internet of Things and its Application to Service Composition*, IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM) (pp. 1-6), IEEE, 2012.

[62] B. Qureshi, G. Min, D. Kouvatsos, *M-Trust: A Trust Management Scheme for Mobile P2P Networks*, IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC) (pp. 476-483), IEEE, 2010.

[63] M. Blaze, J. Feigenbaum, J. Lacy, *Decentralized Trust Management*, IEEE Symposium on Security and Privacy (pp. 163-173), IEEE, 1996.

[64] G. Theodorakopoulos, J.S. Baras, *On Trust Models and Trust Evaluation Metrics for Ad Hoc Networks*, IEEE Journal on Selected Areas in Communications 24(2) (pp. 318-328), IEEE, 2006.

[65] L. Wen, P. Lingdi, L. Kuijun, C. Xiaoping, *Trust Model of Users Behavior in Trustworthy Internet*, WASE International Conference on Information Engineering (pp. 403-406), IEEE, 2009.

[66] J.A. Buchmann, E. Karatsiolis, A. Wiesmaier, *Introduction to Public Key Infrastructures*, Springer, 2013.

[67] The International PGP Home Page, http://www.pgpi.org/

[68] D. Gambetta, *Can We Trust Trust?*, Trust: Making and Breaking Cooperative Relations, (pp. 213-237), 1988.

[69] D. Artz, Y. Gil, *A Survey Of Trust In Computer Science And The Semantic Web*, Web Semantics: Science, Services and Agents on the World Wide Web 5(2) (pp. 58-71),2007.

[70] N.M. Frank, L. Peters, *Building Trust: The Importance of Both Task and Social Precursors*, International Conference on Engineering and Technology Management (pp. 322 - 327), IEEE, 1998.

[71] P. Victor, C. Cornelis, M. de Cock, *Trust Networks for Recommender Systems*, Atlantis Computational Intelligence Systems, Atlantis Press, 2011.

[72] Y. Yu, K. Li, Y. Zhang, L. Xu, *A Service Trust Model with Passive Trust*, IFIP International Conference Network and Parallel Computing (pp. 218-225), 2008.

[73] G. Zhang, J. Kang, R. He, *Towards a Trust Model with Uncertainty for e-Commerce Systems*, IEEE International Conference on e-Business Engineering (pp. 200-207), 2005.

[74] E. Damiani, D.C. di Vimercati, S. Paraboschi, P. Samarati, F. Violante, *A Reputation-based Approach for Choosing Reliable Resources in Peer-to-peer Networks*, Proceedings of the 9th ACM Conference on Computer and Communications Security (pp. 207-216), 2002.

[75] N. Curtis, R. Safavi-Naini, W. Susilo, *X^2Rep: Enhanced Trust Semantics for the XRep Protocol*, Lecture Notes in Computer Science, Vol. 3089 (pp 205-219), 2004.

[76] L. Humphreys, *Mobile Social Networks and Social Practice: A Case Study of Dodgeball*, Journal of Computer-Mediated Communication 13(1) (pp. 341360), 2007.

[77] H. Rheingold, *The Virtual Community: Homesteading on the Electronic Frontier*, MIT Press, 2000.

[78] M. Igbaria, *The Driving Forces in the Virtual Society*, Communications of the ACM, Vol. 42 Issue 12 (pp. 64-70), 1999.

[79] J. Caverlee, L. Liu, S. Webb, *The SocialTrust Framework for Trusted Social Information Management: Architecture and Algorithms, Information Sciences*, Vol. 180 Issue 1 (pp. 95-112), 2010.

[80] H-C. Chen, A. L. P. Chen, *A Music Recommendation System Based on Music Data Grouping and User Interests*, Proceedings of the Tenth International Conference on Information and Knowledge Management (pp. 231–238), 2001.

[81] U.S. Manikrao, T.V. Prabhakar, *Dynamic Selection of Web Services with Recommendation System*, International Conference on Next Generation Web Services Practices, 2005.

[82] M. P. O'Mahony, B. Smyth, *A Recommender System for On-line Course Enrolment: An Initial Study*, Proceedings of the 2007 ACM conference on Recommender Systems (pp. 133-136), 2007.

[83] H. Weihong, C. Yi, *An E-commerce recommender system based on content-based filtering*, Wuhan University Journal of Natural Sciences 11(5) (pp. 1091-1096), 2006.

[84] K. Kimura, H-H. Huang, K. Kawagoe, *Photo-taking Point Recommendation with Nested Clustering*, IEEE International Symposium on Multimedia (pp. 65-68), 2012.

[85] M. Richardson, R. Agrawal, P. Domingos, *Trust Management for the Semantic Web*, The Semantic Web - ISWC 2003 (Second International Semantic Web Conference (pp. 351-368)), Springer Berlin Heidelberg, 2003.

[86] Z. Yan , S. Holtmanns, *Trust Modeling and Management: from Social Trust to Digital Trust*, Computer Security, Privacy and Politics: Current Issues, Challenges and Solutions (pp. 1-28), 2007.

[87] J. Nahapiet,S. Ghoshal, *Social Capital, Intellectual Capital, and the Organizational Advantage*, The Academy of Management Review, Vol. 23 Issue 2 (pp. 242-266), Academy of Management, 1998.

[88] S. C. Craig, *Efficacy, Trust, and Political Behavior: An Attempt to Resolve a Lingering Conceptual Dilemma*, American Politics Research, Vol. 7 Issue 2 (pp. 225-239), 1979.

[89] A. Bandura, *Social Cognitive Theory: An Agentic Perspective* , Asian Journal of Social Psychology, Vol. 2 Issue 1 (pp. 2141), 1999.

[90] R. W. White, M. Richardson, M. Bilenko, A. P. Heath, *Enhancing Web Search by Promoting Multiple Search Engine Use*, Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, 2008.

[91] S. Singh, S. Bawa, *A Privacy, Trust and Policy Based Authorization Framework for Services in Distributed Environments*, International Journal of Computer Science 2(2) (pp. 85-92), 2007.

[92] C. M. Eastman, B. J. Jansen, *Coverage, Relevance, and Ranking: The Impact of Query Operators on Web Search Engine Results*, Transactions on Information Systems (TOIS) 21(4), 2003.

[93] A. Herzberg, Y. Mass, J. Mihaeli, D. Naor, Y. Ravid,*Access Control Meets Public Key Infrastructure, or: Assigning Roles to Strangers*, IEEE Symposium on Security and Privacy (pp. 2-14), IEEE, 2000.

[94] S. Marsh, *Formalising Trust As A Computational Concept*, Ph.D. thesis, Department of Computing Science and Mathematics, University of Stirling, 1994.

[95] C. Hang, M. P. Singh, *Trust-based Recommendation Based On Graph Similarity*, AAMAS Workshop on Trust in Agent Societies, 2010.

[96] R. H. Conviser, *Toward A Theory Of Interpersonal Trust*, Pacific Sociological Review 16(3) (pp. 377-399), 1973.

[97] L. Mui, M. Mohtashemi, A. Halberstadt, *A computational Model of Trust and Reputation*, Proceedings of the 35th International Conference on System Science (pp. 280-287), 2002.

[98] T. Grandison, M. Sloman, *A Survey of Trust in Internet Applications*, Communications Surveys & Tutorials, IEEE 3(4) (pp.2-16), 2000.

[99] D. Olmedilla, O. Rana, B. Matthews, W. Nejdl, *Security and Trust Issues in Semantic Grids*, Proceedings of the Dagsthul Seminar, Semantic Grid: The Convergence of Technologies, 2005.

[100] P. Massa, P. Avesani, *Trust-aware Recommender Systems*, Proceedings of the 2007 ACM conference on Recommender Systems (pp. 17-24), 2007.

[101] P. Massa, P. Avesani, *Trust-aware Collaborative Filtering for Recommender Systems*, Federated Int. Conference On The Move to Meaningful Internet: CoopIS, DOA, ODBASE. Springer-Verlag, 2004.

[102] J. Golbeck, *Computing and Applying Trust in Web-based Social Networks*, Ph.D. dissertation, University of Maryland, College Park, 2005.

[103] D. Gefen, Reflections on the Dimensions of Trust and Trustworthiness Among Online Consumers, SIGMIS Database 33 (3) (2002) 38-53.

[104] R. Levien, A. Aiken, *Attack Resistant Trust Metrics for Public Key Certification*, 7th USENIX Security Symposium, 1998.

[105] S.D. Kamvar, M.T. Schlosser, H. Garcia-Molina, *The Eigentrust Algorithm for Reputation Management in P2P Networks*, Proceedings of the 12th International Conference on World Wide Web (pp. 640-651), 2003.

[106] J. O'Donovan, *Capturing Trust in Social Web Applications*, Computing with Social Trust, pp. 213-257, Springer-Verlag, 2009.

[107] Z. Yan, P. Zhang, R. H. Deng, *TruBeRepec: a Trust-behavior-based Reputation and Recommender System for Mobile Applications*, Personal and Ubiquitous Computing, Vol. 16 Issue 5 (pp. 485-506), 2012.

[108] F. Bustos, J. López, V. Julián, M. Rebollo, *STRS: Social Network Based Recommender System for Tourism Enhanced with Trust*, Advances in Soft Computing, Vol. 50 (pp. 71-79), 2009.

[109] R. R. Sinha, K. Swearingen, *Comparing Recommendations Made by Online Systems and Friends*, DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries, 2001.

[110] M. Jamali, M. Ester, *TrustWalker: A Random Walk Model for Combining Trust-based and Item-based Recommendation*, Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 397-406), 2009.

[111] P. Victor, M. De Cock, C. Cornelis, *Trust and Recommendations*, Recommender Systems Handbook, Springer-Verlag, 2011.

[112] J. O'Donovan, B. Smyth, *Trust in Recommender Systems*, Proceedings of the 10th International Conference on Intelligent User Interfaces (pp. 167-174), 2005.

[113] A. Hofer, W. F. Tichy, *Status of Empirical Research in Software Engineering*, Empirical Software Engineering Issues, Lecture Notes in Computer Science Vol. 4336 (pp. 10-19), 2007

[114] J. Jin, Q. Chen, *A Trust-based Top-K Recommender System Using Social Tagging Network*, 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD) (pp. 1270-1274), 2012.

[115] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, *GroupLens: An Open Architecture for Collaborative Filtering of Netnews*, Proceedings of the 1994 ACM conference on Computer supported cooperative work (CSCW '94) (pp. 175-186), 1994.

[116] M. G. Ozsoy, F. Polat, *Trust based Recommendation Systems*, Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, August, 2013.

[117] J. Golbeck, *Personalizing Applications through Integration of Inferred Trust Values in Semantic Web-Based Social Networks*, In Proceedings of the 4th International Semantic Web Conference (Semantic Network Analysis Workshop), College Park, Maryland, 2005

[118] S. Krishnamurthi, J. Vitek, *The Real Software Crisis: Repeatability as a Core Value*, Communications of the ACM, Vol. 58 Issue 3 (pp. 34-36), 2015.

[119] C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, Prentice Hall, 3rd Edition, 2004.

[120] J. Giribet, M. España, C. Miranda, *Synthetic Data for Validation of Navigation Systems*, Acta Astronautica Vol. 60 Issue 2 (pp. 88 - 95), 2007.

[121] Y. Chen, J. Li, *Generation of Synthetic Data and Experimental Designs in Evaluating Interactions for Association Studies*, Journal of Bioinformatics and Computational Biology Vol. 10 Issue 1, 2012.

[122] M. V. Zelkowitz, *Techniques for Empirical Validation*, Empirical Software Engineering Issues, Lecture Notes in Computer Science Vol. 4336 (pp. 4-9), 2007.

[123] M. A. Whiting, J. Haack, C. Varley , *Creating Realistic, Scenario-based Synthetic Data for Test and Evaluation of Information Analytics Software*, Proceedings of the 2008 Workshop on BEyond Time and Errors: Novel evaLuation Methods for Information Visualization (BELIV), 2008.

[124] B. Franca, G. Travassos, *Are We Prepared for Simulation Based Studies in Software Engineering Yet?*, CIbSE/ESELAW 2012 Special Issue, Vol. 16 Issue 1, 2013.

[125] A. M. Law, W. D. Kelton, *Simulation Modeling and Analysis*, McGraw Hill, third edition, 2000.

[126] D. I. K. Sjoberg, *Knowledge Acquisition in Software Engineering Requires Sharing of Data and Artifacts*, Empirical Software Engineering Issues, Lecture Notes in Computer Science Vol. 4336 (pp. 77-82), 2007.

[127] S. M. Ross, *Simulation*, Academic Press, Inc., fourth edition, 2006.

[128] D. C. Montgomery, G. C. Runger, *Applied Statistics and Probability for Engineers*, John Wiley & Sons, third edition, 2003.

[129] J.H. McDonald, *Handbook of Biological Statistics*, Sparky House Publishing (3rd ed.), Baltimore, Maryland, 2014.

[130] NIST, *Engineering Statistics Handbook*, http://www.itl.nist.gov/div898/handbook/eda/section3/eda35f.htm, last rev.: 2016-07-07.

[131] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-oriented Software, Addison-Wesley Longman Publishing Co., Inc., 1995.

[132] M. Fowler, *Patterns of Enterprise Application Architecture*, Addison-Wesley Longman Publishing Co., Inc., 2003.

[133] I. Robinson, J. Webber, E. Eifrem, *Graph Databases*, O'Reilly Media, Inc., 2013.

[134] M. Hunger, *Querying Graphs with Neo4j*, DZone Refcardz #203, DZone Inc., 2014.

[135] F.E. Ritter, M.J. Schoelles, K.S. Quigley, L.C. Klein, *Determining the Number of Simulation Runs: Treating Simulations as Theories by not Sampling their Behavior*, Human-in-the-loop Simulations: Methods and Practice (pp. 97-116), 2011.

[136] M.D. Byrne, *How Many Times Should a Stochastic Model Be Run? An Approach Based on Confidence Intervals*, In Proceedings of the 12th International Conference on Cognitive Modeling (Canada), 2013.

[137] M.R. Driels, Y.S. Shin, *Determining the Number of Iterations for Monte Carlo Simulations of Weapon Effectiveness*, Report prepared for Defense Threat Reduction Agency, 2004.

[138] N. Chiabaut, C. Buisson, *Replications in Stochastic Traffic Flow Models: Incremental Method to Determine Sufficient Number of Runs*, Traffic and Granular Flow 07 (pp. 35-44), 2009.

[139] W. Burghout, *A Note on the Number of Replication Runs in Stochastic Traffic Simulation Models*, Report: Centre for Traffic Research, 2004.

[140] J.W. Creswell, *Qualitative Inquiry and Research Design: Choosing Among Five Traditions*, Thousand Oaks, CA: Sage Publications, 1998.

[141] B.G. Glaser, A.L. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Piscataway, New Jersey: Transaction, 1967.

[142] J.M. Morse, *Designing Funded Qualitative Research*, Handbook of qualitative research (2nd Ed), Thousand Oaks, CA: Sage, 1994.

[143] T. Gee, *Why Java 8?*, The DZone Guide to the Java Ecosystem, DZone, 2015.