Chapter 43
Extracting Social Relationships from Social Software

Jürgen Dorn
Vienna University of Technology, Austria

Stefan Labitzke
Vienna University of Technology, Austria

ABSTRACT

We present an analytical approach to detect relationships between persons in the real world, such as friendship, rivalry, or others, out of the behavior of members in a social software system. In social software systems, users often evaluate submissions of other users. If these actions are somehow biased, we assume a personal relationship between these users. If we know about the relationship between two users, the validity of the evaluation, and with that, the trust into the social software, can be improved. For example, if a rival evaluates a submission unfairly, we should decrease the impact of this evaluation. We apply the approach in TechScreen, a social software system that supports the exchange of knowledge about Internet technologies. Since we try to mine competencies of its users, the validity of evaluations is very important. In this chapter we show results of experiments with about 50 users.

INTRODUCTION

The use of information systems can be classified in three periods. In the first period, information systems addressed certain tasks of users such as writing texts and other professional tasks at their desktop. The user's task and its correct solution were in the focus. In the second period, the interface between user and computer appeared in the foreground of evaluations. The third period is the move to collaboration between users by means of computers. Such information systems supporting the collaboration are often called social software systems.

Social software (systems) is a class of information systems supporting the establishment and management of on-line communities for people in performing certain tasks. Social software may provide different services for community members such as finding members with similar interests, finding information on interesting subjects, discussing common problems or simply storing

DOI: 10.4018/978-1-4666-4301-7.ch043

of private or publicly accessible documents. By motivating large communities for submissions and structuring the content (e.g. pictures, videos) and making recommendations out of user submissions, the body of the aggregated information achieves considerable worth.

It is only one group of social software that achieves very high volumes of users (e.g. eBay, YouTube, Flickr, Facebook, …). Social software systems are also used to build smaller communities with a restricted access. Thus a company may invite its customers into such a community for online support on products and services of the company. Social software is also used to support knowledge exchange between employees of companies (Wenger 2004). In such "commercial" communities, trust into the system is more important and moreover, due to the relatively few users, the opinion of single users becomes more sensible.

Communities are built around common interests. Often, it is however unclear what the common interests are and whether all community members share the same interests. This problem is also investigated in knowledge management theory. A company should be interested that knowledge is shared between its employees. Knowledge management systems have to be designed in such a way that individual members of the staff are motivated to share relevant knowledge through these systems. Davenport and Prusak (1998) describe three motivations that lead to successful knowledge sharing: reciprocity (if I submit something then also other community members are obliged to share information), reputation (if I submit much information I will be accepted as an expert) and altruism (I want to support this highly relevant community without any immediate benefits). These motivations are also valid for social software applications. A fourth motivation that we want to analyze here, are social relations between users. We assume that some users are motivated by friendship and sometimes there may be also a rivalry relationship that motivates users for negative critics.

The great advantage of a social software system is the free access 24 hours/7 days world-wide supporting asynchronous communication. A number of technologies for indexing, structuring, filtering and recommending were developed in recent years to master large amount of users, their interests and the available information.

In most social software systems users are not verified. Thus a user can pretend to be someone else. Thus a disadvantage in these systems is the missing trust. How can we trust another person/user that we have never met? From social relations we know, that trust grows with the number of fair transactions between two persons. This can also be observed in social software systems. Due to the large number of users, a single user can experience only a relative small number of transactions. However, a system can observe for all users the transaction history and build up trust by considering these transactions.

If many users recommend something, can we trust this "wisdom" of the crowd? Perhaps – but what does it mean if something is only recommended by very few users? If single users rate something as not so good, can we rely on this recommendation? For example, we may find a bad comment on a hotel in an Internet page. This comment may be written by the manager of another hotel who is a rival. And the same manager could write good comments on his own hotel. It is difficult to detect such a biased evaluation. We assume if there is the motivation, then such a biased evaluation will be done regularly, because a single comment would not be so harmful. The question now is how we detect that the same person makes these unfair comments regular. With this method we can increase trust in social software systems.

In the following we regard only social software systems where users have to be registered. What we propose now is to observe the actions of users in the system and to detect how focused their actions are in regard of other users. Thus, it may be that two users react on each other because they have the same interests. The other reason is, that