Determining social relevance in contact-based communication at the example of SNAIL

Qiong Wu
Student at Darmstadt University of Technology
Riedeselsrasse 64 / 7
D-64283 Darmstadt
+49 152 29845748
qiong.wu@studentpartners.de

ABSTRACT
Modern ways of communication embrace contact-based messaging, may it be instant messaging, social networks or telephony services. As the number of contacts increases, a way to determine the relevance of contacts in order to sort contact lists has to be found. Together with SNAIL, an approach to contact-based email usage, the following experiences have been made.

Categories and Subject Descriptors
H.1.2.a Human factors, H.4.3.c Electronic mail

General Terms
Design, Experimentation, Human

Keywords
Email, Communication, Social, Networks, SNAIL

1. INTRODUCTION
Communication seems to be drifting towards contact based communication. No matter if it is Twitter, Facebook, ICQ or Skype, Contact based communication is a winner. But in fact communication is not drifting towards contact based communication, but has always been contact based. Most naturally, in the beginning was the word. And the word was exchanged between individuals. Before mass media conquered our daily lives, communication has always been between individuals. And today one has to distinguish between broadcast media and individual information exchange. We will focus primarily on the latter and try to distinguish genuine communication end-to-end connections from broadcasts which are prevalent in email communication as well.

Peer to peer text communication can be divided into two basic categories: real-time communication and relayed communication. Usually instant messaging is considered real-time as they feature a direct communication with an active peer while email is considered relayed communication as a message is usually sent without knowledge of the activity of the foreign peer.

Email is very special for today's internet communication as it is a very old-fashioned and traditional way to exchange information over the internet but at the same time carries the majority of business information exchange. Almost all of today's enterprise communication is done via E-Mail while instant messaging is often considered as too unserious for business. While some businesses do adapt instant messaging communication for their internal communication, any kind of messaging with clients is solely done via E-Mail.

To compare the up and downturns of both E-Mail and Instant Messaging we sum up the basic features in the following table.

Table 1

Feature	E-Mail	Instant Messaging
Instant Communication	Only with push services, but no knowledge about the online status exists	Fully supported
Offline Messages	Fully supported	Partially available, depending on the protocol, but requires a message cache server
Contact Based	Mails are sent to / received from contacts, but usually are not aggregated by contacts	Usually aggregated in a list, messages are always associated with at least one contact
Message Search	Is considered a major problem for email as old email is often hard to find	Usually very convenient way to search through a contact's message history

It is easily observable that instant messaging holds some clear benefits over email. But on close observation it is revealed that the problems do not originate from the protocol design of email but actually from the design of email applications.

Numerous additions and tweaks have been developed to provide a more instant messaging like experience of email such as the

blackberry push service or numerous attempts from google to change the way we email. While those attempts were more or less successful, they did not target the problems that SNAIL targets.

The basic idea of SNAIL is a completely contact based way of writing and reading emails as a supportive way to handle the enormous masses of emails the average email user has to handle. SNAIL tries to take care of exactly that information exchange between individuals by extracting contacts from the heap of emails received and delivering them in a convenient way we're used from applications like MSN, ICQ or Skype, the contact list.

What seems ridiculous and/or revolutionary at first glance is actually just basically applying instant messaging principles to email communication. It is very trivial because instant messaging is in the first place very similar to email. Both technologies support multiple recipients (IM not always), and both offer a way to connect messages by threads (although usually not recognized when used with IM). There is little significant difference between the technologies on the network layer and disparities only occur at the way they present the information to the user. The fact that email usually utilizes server pull instead of server push to retrieve messages is not of significant importance to SNAIL, while the lack of an online status in the email design (due to the modeling after real snail mail) is a problem that has to be somehow approached by SNAIL.

There have been numerous efforts [2],[3] to introduce instant messaging to the enterprise environment, but unfortunately the majority failed because of network security issues or unwillingness of users to adapt to the new communication model. Studies have shown that while private users readily embrace instant messaging, businesses face serious problems when trying to introduce instant messaging to their employees.

It should be noted though, that even though SNAIL introduces elements of instant messaging to email communication, the underlying way of communication is not altered. Instant messaging differs from email significantly in the way users communicate, as messages are usually much shorter and conversations have a much higher reply rate. SNAIL does not introduce this to the email communication.

2. User Interface Design

SNAIL has been developed as an outlook plugin in order to seamlessly integrate into everyday email usage. The development of an plugin on top of an existing email client makes sense, as the features SNAIL provides are extensions to the way we read email. The common way of reading email is not rendered useless by SNAIL. In addition, by providing an email client plugin, existing users are more motivated to give SNAIL a try, as this does not interfere with everyday email usage as much as the introduction of a completely new standalone tool would.

To develop a **person centered** way of interacting with email, SNAIL provides a instant messaging styled contact list as main user interface. In addition the user can view the history of a contact in the list by invoking the history view window. Both concepts are very common to instant messaging, especially to clients like ICQ, MSN or Skype and are the de-facto standard for real-time peer to peer message communication.

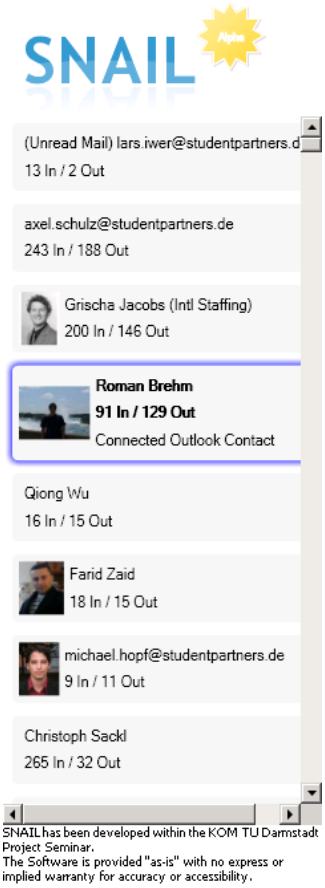
In Figure 1 a typical instant messaging application is shown while in Figure 2 the SNAIL contact list is shown in comparison.

2.1 Contact List

The contact list is populated with all contacts known to the user, which is determined by getting all recipients and senders from all email items in outlook. When an entry is selected, detailed information about the contact, such as a connected outlook or facebook contact, is displayed.

Figure 1: IM Contact List

Figure 2: SNAIL Contact List



Depending on the status of the contact, actions may be performed such as

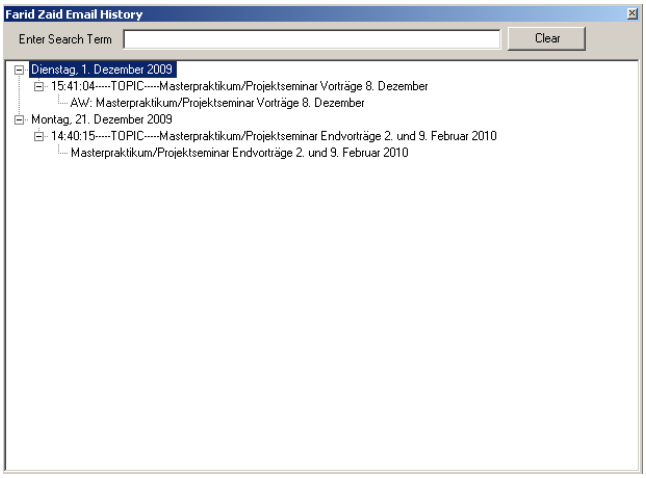
- Writing a new Email
- Opening Unread Email (if existent)
- View the conversation history
- Associate contact with outlook or facebook contact

The way of interaction with the contact list is strictly modeled after instant messaging contact lists, except for that due to a lack of online status indication, contacts cannot be sorted by status. This would also not be feasible for the email scenario, as a user is not more likely to write an email to a contact just because he is available at the moment. Here the disparity of use case between email and instant messaging comes into play. While instant messaging is typically used for chatting, email is primarily used for formal information.

2.2 The History View

For the visualization of conversation history a classic tree concept is used. The messages are aggregated by date and then by conversations. A double click on a message entry opens the corresponding email in a new outlook window.

Figure 3: SNAIL Thread View



The history tree can be searched by entering a search term, all message subjects matching the search terms are then displayed in a filtered view.

3. Approach

3.1 Evaluated Data

In order to make conclusions for SNAIL, information data has to be extracted from outlook.

The important fields for data evaluation are not specific to outlook, but to email in general.

Important Email Fields

- Subject
- Sender
- Recipients
- Received / Sent Date
- Conversation Thread

While the email body may be of interest, the expense for saving and processing has been deemed too high in comparison to the information which could possibly extracted from the email body. In general it is assumed that the subject of an email is explanatory to the content it contains.

Extracted Information for evaluation

- Shared Contacts
- Sender
- Amount of Emails sent to / received from a contact
- Received / Sent Date
- Conversation Thread start date

Particularly the shared contacts are of interest. They are generated by assuming that a contact who is part of a recipient list of an email shares contact with all contacts who are on the recipient list and the sender as well.

3.2 Determining the relevance

When a contact list has been created, the question after some kind of sorting algorithm arises quite naturally, as the number of contacts generated from a mailbox is usually quite overwhelming. Obviously an alphabetical sorting order is not feasible for the contact list. For instant messaging services the answer is very trivial, because instant messaging contacts possess an online/offline status. Naturally one would primarily want to write a message to a contact which is available at the moment. But for email based communication there is no such thing as an online status. Instead we have to rely on other factors to determine which contacts are most relevant to the user. Keep in mind that the user probably wants to have those contacts on top that he wants to talk to most.

We subsequently call these contacts **natural contacts** as they are contacts that map to a real person contact, not a mailing list or a bot.

But how can we determine this? Different approaches can be done here. Evaluating different approaches with user experience tests yield the best possible solution for the problem.

After extracting the raw data and storing it programmatically, we can use the information provided by the raw data to perform sorting on our contacts.

Possible solutions contain evaluating

- Counting the amount of characters/words/messages exchanged with the contact.
- Determining the duration of threads in which the user and the contact participated
- Counting the amount of initial messages sent or received from the contact
- Scanning the mail content for relevant keywords

All possibilities can also be evaluated in multiple different ways. An averaged metric can be used in opposition to an absolute metric. Also it would be important to determine whether a 3-month evaluation is more feasible or an infinite duration evaluation.

Although multiple sophisticated methods for email trust & relevance determination exist, a simple approach has been deemed more feasible as an exact determination of contact importance is not necessarily needed. Instead we want to find the contacts that the user would most probably write a message to.

In this context the ration of in/out messages is also especially important. Usually contacts which only send emails to the user can be expected to be some kind of newsletters / bots or mailing lists, although predictions can only be done after a significant number of messages are received from this contact in order to avoid wrong ordering of new contacts.

To counter false sorting of new contacts (contacts with 1 received mail) which would result in a bad in/out ratio, contacts with unread messages are always sorted on top of the list. If the user reads the mail and replies, the score is evened out (1/1).

The sorting of contacts with unread messages to the top also provides a good way to deal with email overflow, as all contacts with unread mails are sorted by their corresponding relevance.

Basically the list is split into a partial list with unread mail contacts and common contacts. Both partial lists are sorted inherently by their relevance then, so that if a user receives a lot of emails he is still able to distinguish important from unimportant emails. To tackle email overflow with SNAIL the user can simply double click a contact in the list to open all unread mails of the contact.

3.3 Utilizing external sources

In addition to relevance calculations done solely on the data gathered by the email plugin, external sources can or even should be used as well. Social networks like Facebook or LinkedIn can be used to determine the relevance of a contact. Xobni is an example for an email based tool which is able to determine social relationships based on social networks. The idea is simple; if a contact can be found within the contact list of a user on Facebook it is very probable that the contact is relevant to the user. An acknowledgement from the user might be requirement in order to avoid wrong predictions.

For SNAIL, facebook and outlook contacts are taken into account and can be associated with contacts in the contact list. The user manually has to associate the contacts in the list with outlook or facebook contacts. For the evaluation of contact relevance this factor could be used as it could be assumed that contacts with a social network connection are of importance to the user.

3.4 Correlation of relevance factors

Different relevance factors have to be taken into account when finally sorting the contact list. For this purpose a scoring system has been developed, in which different factors of relevance yield different amounts of "points". The initial value is set to zero which corresponds to a newly added contact. From this position points can either be gained or lost, leading to either positive or negative scores. Negative contacts are sorted to the rear of the list; positive contacts are sorted to the head of the list.

3.4.1 Concept One

Concept One has been developed to provide a basic model to determine which contacts are relevant. It aims at giving contacts a high rank whose Sent/Received ratio is close to 1 (balanced). The idea is that natural contacts usually have a real conversation with the user (a message is sent, it is replied to, etc.).

To rank heavier conversations higher (e.g. to avoid 1/1 or 2/2, X/X conversations to always rank highest) the score is weighted by the amount of emails sent to the user.

This way mailing lists or bot notifications will not receive a high rating as the user is not likely to reply to those.

Score determination

$$\text{Score} = \frac{\left|1-\frac{Sent+0{,}00001}{Received+0{,}00001}\right|+0{,}1}{Sent+0{,}00001}$$

- Unread Mails
- Score

To avoid division by zero, an arbitrary small value is added to the denominator

3.4.2 Concept Two

Another approach has been tested in which a sorting is done by taking the date of the last conservation into account.

- Unread Mails
- Last Conservation
- Amount of Sent messages.

3.4.3 Optimizing models

While the first concept does not take time into account at all, the second concept sorts the contact by the last conversation as well. During development it has shown that by just sorting by the last conversation date an unrealistic sorting order would be achieved, as contacts who the most recent conversation date would not necessary be contacts that the user would be interested in.

Instead the first concept had been enhanced in order to take time into account. By only including emails sent or received in a certain time window (x days from now) for the calculation of the score, a very realistic sorting order could be achieved. By giving the user the possibility to set the timespan manually a high grade of customization could be achieved. As for the default setting, 30 days has proven to be a good approximation for the perfect sorting timespan.

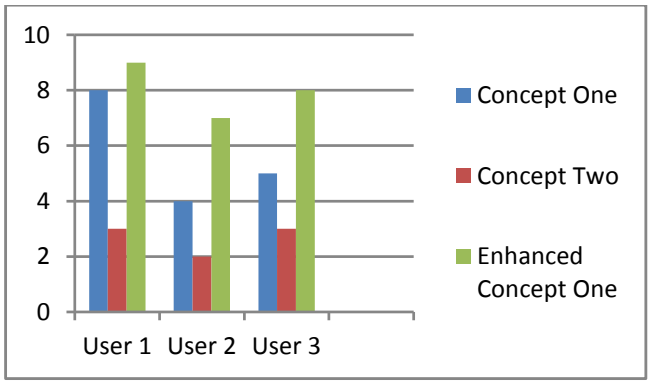
It should be mentioned that social network connections have not been taken into account for both concept as it turned out that social network connections were not necessarily a factor of relevance, especially in business environments where important people are rarely included in a social network. A trial with inclusion of social network factors resulted in a bad experience of the sorted contacts.

4. Experiments

By setting different priorities when sorting contacts by relevance and then comparing to actually known relevance, the best method has been determined by our tests.

A subject chooses 10 contacts he most frequently contacts and then runs SNAIL to generate a presorted contact list. The amount of chosen contacts appearing in the top 10 of the presorted list is a measurement value for the quality of our sorting algorithm.

Due to limited time & resources we did a test with 3 independent users with User 2 & User 3using their email accounts primarily for private mailing while User 1 ran SNAIL over his corporate mail account.



The discrepancy between Concept One and the Enhanced Concept One is particular interesting, as it is much lower for the corporate

Table 2: Sample Evaluation of Concept One

Sent	Received	Interpolated Ratio	Interpolated Deviation	Average	Score	Real Ratio	Real Deviation	Real Deviation [%]
123	145	0,848275873	0,251724127	134	0,002046537	0,848275862	0,151724138	15,17%
91	129	0,705426379	0,394573621	110	0,004335973	0,705426357	0,294573643	29,46%
20	20	1	0,1	20	0,004999998	1	0	0,00%
16	15	1,066666622	0,166666622	15,5	0,010416657	1,066666667	0,066666667	6,67%
15	16	0,937500039	0,162499961	15,5	0,010833324	0,9375	0,0625	6,25%
3	3	1	0,1	3	0,033333222	1	0	0,00%
345	23	14,99999391	14,09999391	184	0,040869546	15	14	1400,00%
23	345	0,066666694	1,033333306	184	0,044927516	0,066666667	0,933333333	93,33%
15	200	0,075000046	1,024999954	107,5	0,068333285	0,075	0,925	92,50%
1	1	1	0,1	1	0,099999	1	0	0,00%
2	3	0,666667778	0,433332222	2,5	0,216665028	0,666666667	0,333333333	33,33%
2	1	1,99999	1,09999	1,5	0,54999225	2	1	100,00%
1	1000	0,00100001	1,09899999	500,5	1,098989	0,001	0,999	99,90%
1	0	100001	100000,1	0,5	99999,10001	#DIV/0!	#DIV/0!	#DIV/0!
67	0	6700001	6700000,1	33,5	99999,98657	#DIV/0!	#DIV/0!	#DIV/0!
0	40	2,5E-07	1,09999975	20	109999,975	0	1	100,00%
0	130	7,69231E-08	1,099999923	65	109999,9923	0	1	100,00%

email user. The explanation may be that in a business environment, contacts are less likely to change a lot, as the most important contacts usually are colleagues or superiors, while private users have a higher fluctuation of contacts. Unfortunately the discrepancy may also be only due to the low number of samples for our tests, further evaluation on this matter is therefore crucial and should be performed on further analysis.

4.1 Results

Concept one has shown to work well with normal scenarios but in theory might have the disadvantage of miss-rating contacts with high conversation flow but no recent messaging. Concept Two has experienced some weaknesses as we found out that contacts that we recently had conversations with were not necessarily as important to us as heavyweight conversation partners.

The enhanced version of concept one has proved to be very accurate as to the determination of relevance of a contact.

Testing of SNAIL on three different mailboxes resulted in a very positive feedback from users who reported that SNAIL enhanced their email experience when

- Finding the contact when writing a new mail
- Looking for a specific mail sent by/to a specific contact

When asked about the sorting of the contact list, users responded that they were surprised by how accurate SNAIL presents natural contacts at the top of the list but nevertheless criticized that there was no way to manually sort a contact up or down.

5. Related work

Another proof of concept has been done by Microsoft Research with the semi-standalone Email client SNARF [1] which makes a similar aggregation of contacts and provides a customizable relevance sorting to the user. E.g., the user can select a sorting by all mails received divided by the amounts of mail sent in the last 30 days. Basic Mathematical operations may be performed and a time correlation can be done. SNARF has been retired from active development as the designated goal of solving the problem of email overflow was not achieved. The experiences with SNARF were used in SALSA [4], another Microsoft Research project which is also realized as an Outlook plugin. In addition, SALSA is part of a standalone social network where managed contacts are organized and separate connections within the network can be made.

One substantial difference between SNARF and SNAIL is that SNAIL integrates seamlessly into the existing MS Outlook environment. On the whole, SNAIL seeks to provide a high level of simplicity to the user. For example no significant customization of the user interface can be done, while SNARF provided a multitude of ways to customize the way emails were presented to the user which supposedly led to some confusion among users.

SALSA seems to follow a more simple way to present email aggregation to the user as it integrates seamlessly into outlook and has a more user friendly interface. Unfortunately SALSA has not been released to public and only little information about it has been published. The results of SALSA have contributed to the social connector feature of the upcoming Outlook 2010.

The enhancement of email is being developed by the big players in the industry right now, not only Microsoft is looking for ways to make email more convenient, Google is constantly looking for ways to improve email experience too. While the focus is right now especially on linking social networks (and their services) to email clients (Google Buzz), email providers still seek ways to keep email competitive in contrast to social network messaging & real-time chats. The integration of Google Chat to Google Mail is just one example.

Nevertheless, the simple concept of SNAIL has not yet been introduced by email client developers, SNARF and SALSA may incorporate similar principles, but lack the basic idea of a contact list with instant messaging mimic.

On the topic of relevance & trust determination in context of email networks a lot of research has been done with more or less successful results. In most cases the evaluation of data has been very complex, taking social connections into account (shared

contacts), weighting them by CC/To/BCC and then calculating a graph which then determines the relevance of the contact to the user. Often such systems are meant to run on distributed systems, gathering data from each client respectively and then correlating them which each other. While this may prove very useful for spam detection & filtering (by analyzing email traffic on servers instead of analyzing emails on the client), such an accurate determination of trust comes usually with high cost and is often not realizable on a single user system.

6. Conclusions & Outlook

The result of the measurements provided us with a promising future for hybrid email communication, empowering business wide used email services with the capabilities and comfort of instant messaging.

As for the sorting techniques introduced, a very satisfactory result was achieved which yielded a simple, yet effective way to sort contacts, such that the user is provided with a convenient way to access most important contacts.

SNAIL provided a good demonstration and proof of concept for an instant messaging style user interface built on top of email.

It has incorporated known instant messaging paradigms to classic email communication in order to approach the problems of today's email communication.

Further evaluations on sorting mechanism could further improve the experience of SNAIL. Nevertheless, an in-depth analysis of the way users work with SNAIL would be of enormous use in the development of new techniques for SNAIL.

7. ACKNOWLEDGMENTS

I would like to thank all my friends who voluntarily tested this software and Farid Zaid and Mathias Kropff from the KOM department at Darmstadt University of Technology for their support on my project.

8. REFERENCES

[1] Carman Neustaedter, and A.J. Bernheim Brush, Marc A. Smith, 2005. The Social Network and Relationship Finder: Social Sorting for Email

[2] A.F. Cameron, J. Webster / Computers in Human Behavior 21 (2005) 85–103

[3] Ellen Isaacs, Alan Walendowski, Steve Whittaker, Diane J. Schiano & Candace Kamm / The Character, Functions, and Styles of Instant Messaging in the Workplace

[4] Kristin Stecher, Scott Counts, Lili Cheng, Shane Williams, Andrzej Turski / Salsa: Leveraging Email to Create a Social Network for the Enterprise

[5] Andrzej Tursk, Debbie Warnack, Lili Cheng, Shelly Farnham, Susan Yee / Inner Circle – People Centered Email Client

[6] Bonnie A. Nardi , Steve Whittaker , Ellen Isaacs , Mike Creech , Jeff Johnson , John Hainsworth, Integrating communication and information through ContactMap, Communications of the ACM, v.45 n.4, p.89-95, April 2002 [doi>10.1145/505248.505251]