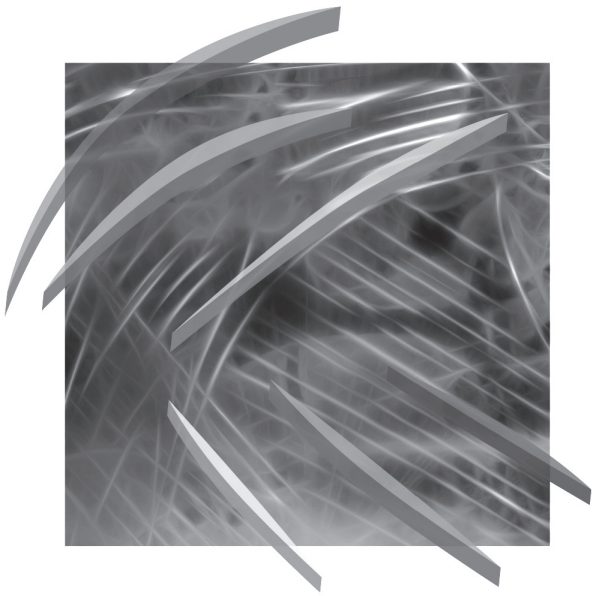
# INFORMATYKA EKONOMICZNA
## BUSINESS INFORMATICS

21 • 2011



Publishing House of Wrocław University of Economics
Wrocław 2011

Copy-editing: Agnieszka Flasińska, Elżbieta Macauley, Tim Macauley,

Layout: Barbara Łopusiewicz

Proof-reading: Barbara Cibis

Typesetting: Małgorzata Czupryńska

Cover design: Beata Dębska

This publication is available at www.ibuk.pl

Abstracts of published papers are available in the international database The Central European Journal of Social Sciences and Humanities http://cejsh.icm.edu.pl
and in The Central and Eastern European Online Library www.ceeol.com

Information of submitting and reviewing papers is available on the Publishing House's website www.wydawnictwo.ue.wroc.pl



**ISSN 1507-3858** (Business Informatics)
**ISSN 1899-3192** (Research Papers of Wrocław University of Economics)

The original version: printed

Printing: Printing House TOTEM
Print run: 200 copies

## Contents



## Streszczenia

INFORMATYKA EKONOMICZNA BUSINESS INFORMATICS 21 • 2011

ISSN 1507-3858

**Jakub Swacha**
Higher Vocational State School, Gorzów Wielkopolski, Poland
jakubs@wneiz.pl

# AN E-MAIL EXCHANGE ANALYSIS FRAMEWORK FOR PROJECT MANAGEMENT SUPPORT

**Abstract:** In the paper a new e-mail exchange analysis framework is proposed and described, aimed at supporting project management in both business and non-business organizations. The framework acquires data from IMAP servers and processes them to create reports on communication activity between project stakeholders in graphical and tabular forms. The reports can be useful in analyzing the internal communication structure of the project team, identifying the key project members responsible for communication with external parties, as well as discerning project members with over- or sub-average communication activity. The framework has been implemented in Python as open-source software.

**Key words:** e-mail, project management, analysis framework.

## 1. Introduction

### 1.1. Motivation

The implementation of every project requires active communication, both between project participants and with external parties. Monitoring and analyzing the communication in a project can constitute a rich source of information useful for controlling and coordinating the project. Communication problems (such as lack of communication, wrongly directed communication, profuse dissemination of information) can harm the development of any project, regardless of its size, they may still be especially difficult to cope with in the case of larger projects, where they are not only more probable to appear but also harder to diagnose.

Even though there are various means of communication available to be used during project realization, electronic mail (email) remains one that is often and widely used. The digital form of its messages, along with its metainformation structure being precisely defined by IETF standards, make it easily accessible for monitoring and analyzing. With a large amount of email messages exchanged even in small projects, the analysis can be performed in practice only with the help of software tools. The problem is that there are no ready-to-be-used tools capable of reporting information

on email exchange in a scope and form most useful from the project management perspective.

### 1.2. Problem setting

Project management can benefit from various kinds of information that can be obtained from an analysis of email exchange. Presumably the most valuable of them are:

– the structure of the communication within a project, indicating the senders, recipients, links between them, and the intensity of email exchange,
– personal communicational activity, indicating the share of email exchange attributed to each project stakeholder,
– temporal communicational activity, indicating the share of email exchange attributed to periods of time during project realization.

The purpose of the proposed framework is to provide each of the three kinds of information mentioned above in a form that is most adequate both for the structure of the respective kind of information and its further usage, considering a human as its direct receiver (rather than further automatic processing).

### 1.3. Approach

The framework uses an Internet Message Access Protocol (IMAP) [Crispin 2003] server mailbox as its data source. This approach has many advantages over two other possible solutions (gathering data by monitoring traffic at the server, or connecting to an email client, such as, e.g. Microsoft Outlook). The main advantages are as follows:

– data may be collected from multiple sources: different mailboxes at different servers, possibly outside of the project-hosting organization, only provided they are IMAP-compatible (if traffic had been monitored at some server, all the traffic would have to go through that server),
– data may be collected regardless of the email client project stakeholders use: the data are acquired from the server, not the client (if data had been acquired from a certain type of an email client, all the project stakeholders would have to use it),
– data may be collected from periods predating the decision of collecting them: one can analyze any period of project realization provided the project participants were using IMAP protocol for their email correspondence and did not flush their mailboxes (actually, most servers offer message undelete function, which in many cases may recover messages that were already deleted) (monitoring traffic at the server can only register communication occurring at present),
– collecting data requires user login, there is no problem with privacy infringement in the form of collecting data that were not supposed to be collected (which could happen when monitoring traffic at the server).

The framework collects data from a server in a single run then stores them off-line. In this way, the actual analysis may be performed many times without creating an additional load on the mail server. The data stored off-line consist only of elements necessary for the analysis (communication participants, date and time, bytes transferred, and no message subject nor content), which vastly reduces the risk of leaking project-critical data.

## 2. Related work

The research on email mining has been greatly fuelled by the seminal paper of Wellman on social networks based on computer networks [Wellman 2001]. R. Guimera et al. examined email exchange in a real-world organization to determine its community structure [Guimera et al. 2002]. In 2003, J.R. Tyler et al. described a method for the automatic identification of communities of practice from email logs within an organization [Tyler, Wilkinson, Huberman 2003]. In the same year EmailNet was developed: a system that automatically mines the organizational email traffic and generates information on social networks for further analysis [Van Alstyne, Zhang 2003]. Another system, presented by A. Culotta et al., extracts a user's social network and its members' contact information given the user's email inbox, and is capable of identifying unique people, finding their Web presence, and automatically filling a contact address book [Culotta, Bekkerman, McCallum 2004]. In 2006 T. Mitchell et al. presented a clustering approach to automatically acquire information on users' activities by analyzing the data stored at their workstations, including their emails [Mitchell et al. 2006]. Also, Ch. Bird et al. described results of their work on mining the email social network on the Apache HTTP server project, also including automatic resolving of multiple email aliases used by the same individuals [Bird et al. 2006]. M. Studer et al. used KDE email archives to build a typology of so-called participation trajectories in order to achieve a better understanding of an open source software social structure [Studer, Müller, Ritschard 2007]. Recently, D. MacLean et al. proposed an algorithm for creating social topologies (sets of potentially overlapping and nested social groups that represent the structure and content of a person's social network) by mining communication history and identifying likely groups based on co-occurrence patterns [MacLean et al. 2011].

Regarding usage of the email mining results, A. Perer and M.A. Smith proposed three visualizations that capture hierarchical, correlational, and temporal patterns present in a user's email, promoting them as valuable for reflection on social relationships and communication history [Perer, Smith 2006]. D. Fisher et al. showed that social network metainformation can be used as an aid in email triage process, as implemented in the SNARF Outlook add-in module [Fisher et al. 2007], whereas Sh. Yoo et al. used it to automatically prioritize received messages according to the specific priorities of each user [Yoo et al. 2009]. M. Laclavik et al. discussed benefits of the analysis of enterprise and personal email archives, such as the better

understanding of email content and allowing various types of applications to work within a scope of an organization or a community [Laclavik et al. 2010].

## 3. Framework components

The general concept behind the proposed framework is that it should be easy-to-use, with very little effort needed from a user both for preparing and performing the analysis. The framework components are implemented as stand-alone applications with parameterization limited to the necessary minimum. The main framework components are: Mail Fetch, Mail Graph, Mail Table, and Mail History.

### 3.1. Mail Fetch

Mail Fetch is an application responsible for retrieving from a given mailbox all the raw data needed for further analysis, and saving them in a local file. The following input parameters are required to be specified by the user (either as program execution parameters or input using text console after application is started, user's choice): IMAP server address, login, password (can only be input after application is started), mailbox name, and the begin date of the analyzed period. The following input parameters may optionally be specified as program execution parameters, otherwise the default names are used: name of the input file, containing a list of contacts relevant to the project (*addresses*), and name of the output file, containing retrieved data (*emails*).

The *addresses* file should be a Comma-Separated Values (CSV) file using a semicolon as field separator and new line character as record separator. Its records should contain the following fields: Contact short name (initials are strongly suggested), Contact long name (full name of person or institution), Contact class (such as, e.g. different project teams, external contractors, etc.), Primary email address. Those fields can be followed by any number of optional fields containing Alternative email addresses.

The *emails* is a CSV file using a comma as field separator (the difference with the *addresses* file is to help avoid misplacing them) and new line character as record separator. Its records contain the following fields: Message ID (notice that this is the ID returned by the server and thus it is not guaranteed to be unique!), Date and time the message was sent (in the *YYYY-MM-DD HH*:*MM* format), Sender short name (initials), Recipients short names (sequence of initials separated with spaces), Message length (including attachments, in bytes).

Mail Fetch is accompanied by two additional tools. List Folders lists IMAP mailboxes at the given email account (as their names often differ with what email clients display). Get Email Addresses lists all the contacts (address, names) mentioned in message headers in a given mailbox during a specified time (it is useful if the *addresses* file cannot be created based solely on the project documentation or the address book of the email client).

### 3.2. Mail Graph

Mail Graph is an application supporting the analysis of the project communication structure. There are no required input parameters, the following parameters may optionally be specified as program execution parameters, otherwise the default names are used: name of the output file containing mail exchange graph visualization (*graph*), name of the input file containing a list of contacts relevant to the project (*addresses*), name of the input file containing data retrieved by Mail Fetch (*emails*).

The output file *graph* is in Scalable Vector Graphics (SVG) format, and as such can be viewed in any SVG-supporting browser (most contemporary web browsers support it). The graph visualization shows project stakeholders as graph vertices (the shape and colour of each vertex depends on the class of the respective project stakeholder). Each vertex is marked with a contact short name (a contact long name is displayed after hovering the mouse over a vertex in a browser). The contact class and names are taken from the *addresses* file. The edges represent email exchange between project participants: an edge is drawn if at least one mail has been sent from one to another (notice that the graph is undirected). The distance between vertices depends on edge weights (the larger the weight, the smaller the distance), which in turn are calculated with the following formula:

$$w(v_a, v_b) = \left\lfloor \log_2(m_{a,b} + m_{b,a} + 1) \right\rfloor,$$

where: $v_a$ – the graph vertex representing project stakeholder *a*,
$m_{a,b}$ – the number of messages sent from the project stakeholder *a* to the project stakeholder *b*.

### 3.3. Mail Table

Mail Table is an application supporting the analysis of the communicational activity of individual project stakeholders. There are no required input parameters, the following parameters may optionally be specified as program execution parameters, otherwise the default names are used: name of the output file containing the email exchange table (*tab*), name of the input file containing a list of contacts relevant to the project (*addresses*), name of the input file containing data retrieved by Mail Fetch (*emails*).

The output *tab* file is in HyperText Markup Language (HTML) format, and as such can be viewed in any HTML-supporting browser. The file contains three tables:

– Message Count Table, showing the number of messages exchanged between project stakeholders (senders are listed in columns, recipients in rows),
– Message Volume Table, showing the data volume (in mebibytes) exchanged between project stakeholders (senders are listed in columns, recipients in rows),

– Top Senders, listing the message senders sorted by number of messages, and showing the number of recipients, messages, data volume and number of messages sent to respective recipients sorted in descending order by this criterion (the share in percent is displayed after hovering the mouse over a number in a browser).

### 3.4. Mail History

Mail History is an application supporting the analysis of the temporal communicational activity within a project. There are no required input parameters, the following parameters may optionally be specified as program execution parameters: name of the output file containing a histogram of mail activity in time (*history*), name of the input file containing a list of contacts relevant to the project (*addresses*), name of the input file containing data retrieved by Mail Fetch (*emails*). If the *history* file name is omitted, no output file is generated, but the histogram is displayed on screen; in the case of the two remaining parameters, the respective default names are used.

The output *history* file is in SVG format, and as such can be viewed in any SVG-supporting browser. The histogram shows as many bars as there were days in the period under analysis, however, for the sake of clarity, in the case of periods longer than 100 days, only 100 bars are displayed (thus, each bar represents a time span longer than one day). The height of each bar is proportional to the number of messages exchanged during the respective time span. The bars are multi-coloured: each colour indicates a different class of sender, and the size of the bar's part in that colour indicates the relative contribution of that project stakeholder class to the overall mail activity in the respective time span.

## 4. Implementation

The framework has been implemented in Python version 2.6.1. Python has been chosen due to its very high level nature, which allowed high productivity (a few lines of program code are often enough to perform complex tasks) and thus, fast development. The following open-source Python extension modules were used in framework components:

– ProcIMAP version 2.0 (from http://github.com/goerz/procimap) in Mail Fetch,
– igraph version 0.5.4 (from http://igraph.sourceforge.net) in Mail Graph,
– matplotlib version 1.0.1 (from http://matplotlib.sourceforge.net) in Mail History.

Mail Graph uses Fruchterman-Rheingold algorithm for the graph layout [Fruchterman, Rheingold 1991].

## 5. Example of real-world application

The framework has been used in analysis of email exchange in Baltic Museums 2.0 Online Information Platform development project (www.balticmuseums.org). It provided valuable information on the structure of email exchange related to the project, communication habits of certain project participants, as well as changes in communication activity level during project realization. Whereas graphs and histograms produced by framework gave a general glimpse of the mail exchange, the tables allowed for more detailed analysis of the subject. The results of the analysis provided the base for several team organization improvements.

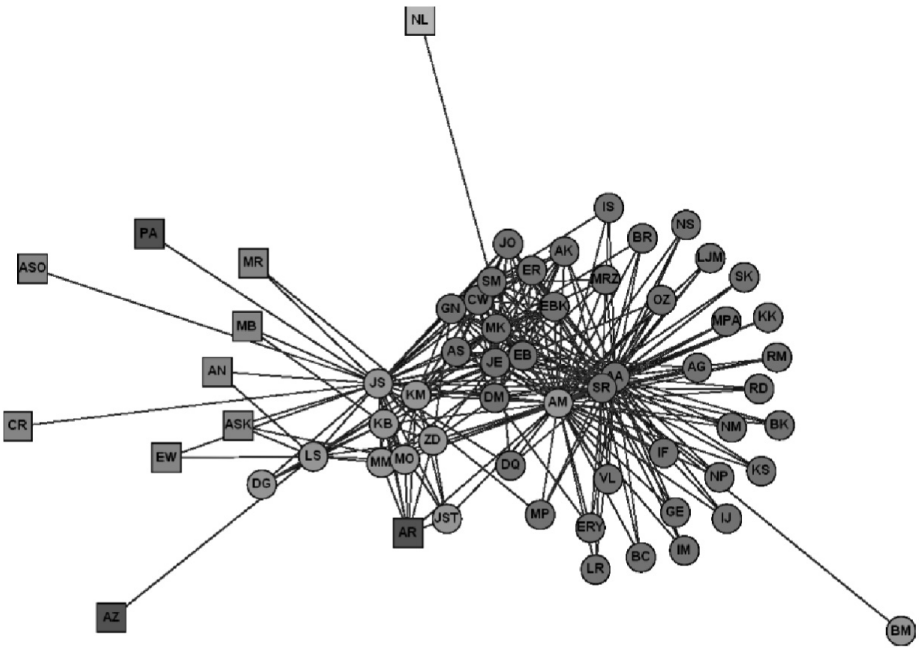



**Figure 1.** The email exchange network of the project development team

Source: own elaboration.

Figure 1 shows the email exchange network of the project development team obtained using Mail Graph. Different shapes and shades of gray represent different classes of project stakeholders.

Figure 2 shows the communication activity related to the project development in subsequent periods of time, obtained using Mail History. Different shades of gray represent different classes of project stakeholders.

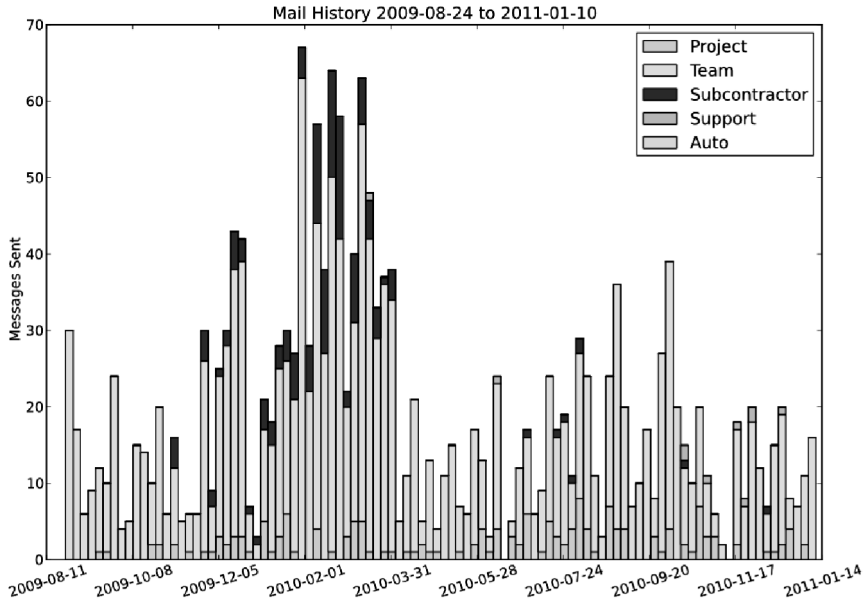



**Figure 2.** The email exchange during project development

Source: own elaboration.

## 6. Summary

This work contributes to the research on the analysis of information exchange from the perspective of project management in both business and non-business organizations. The described framework automatically prepares three types of reports conveying different types of email exchange data. It is an easy to use solution, provided as open-source software, which can be freely downloaded from http://uoo.univ.szczecin.pl/~jakubs/mxa.

## References

Bird Ch., Gourley A., Devanbu P., Gertz M., Swaminathan A., Mining email social networks, [in:] *Proceedings of the 2006 International Workshop on Mining Software Repositories*, ACM, Shanghai 2006, pp. 137–143.

Crispin M., *RFC 3501 – Internet Message Access Protocol – Version 4rev1*, IETF, 2003, http://tools.ietf.org/html/rfc3501.

Culotta A., Bekkerman R., McCallum A., Extracting Social Networks and Contact Information from Email and the Web, [in:] *First Conference on Email and Anti-Spam*, CEAS, Mountain View 2004, http://www.ceas.cc/papers-2004/176.pdf.

Fisher D., Brush A. J., Hogan B., Smith M., Jacobs A., Using social metadata in email triage: Lessons from the field, *Lecture Notes in Computer Science* 2007, Vol. 4558, pp. 13–22.

Fruchterman T.M., Rheingold E.M., Graph drawing by force-directed placement, *Software – Practice & Experience* 1991, Vol. 21 (11), pp. 1129–1164.

Guimerà R., Danon L., Díaz-Guilera A., Giralt F., Arenas A., *Self-similar Community Structure in Organizations*, 2002, arXiv:cond-mat/0211498v1 [cond-mat.dis-nn].

Laclavik M., Kvassay M., Dlugolinsky S., Hluchy L., Use of email social networks for enterprise benefit, [in:] *International Workshop on Computational Social Networks, International Conference on Web Intelligence and Intelligent Agent Technology*, IEEE/WIC/ACM, Toronto 2010, http://laclavik.net/publications/email_SN_final_poster4.pdf.

MacLean D., Hangal S., The S.K., Lam M.S., Heer J., Groups without tears: Mining social topologies from email, [in:] *Proceeding of the 15th International Conference on Intelligent User Interfaces*, ACM, Palo Alto 2011.

Mitchell T., Wang S., Huang Y., Cheyer A., Extracting knowledge about users' activities from raw workstation contents, [in:] *Proceedings of the 21st National Conference on Artificial Intelligence*, Vol. 1, AAAI, Boston 2006, pp. 181–186.

Perer A., Smith M.A., Contrasting portraits of email practices: Visual approaches to reflection and analysis, [in:] *AVI '06: Proceedings of the Working Conference on Advanced Visual Interfaces*, ACM, New York 2006, pp. 389–395.

Studer M., Müller N.S., Ritschard G., Understanding the KDE Social Structure through Mining of Email Archive, [in:] *2nd Workshop on Public Data about Software Development, Third International Conference on Open Source Systems, IFIP*, Limerick 2007, http://mephisto.unige.ch/pub/publications/gr/wopdasd_studer_et_all_full.pdf.

Tyler J.R., Wilkinson D.M., Huberman B.A., Email as spectroscopy: automated discovery of community structure within organizations, [in:] M. Huysman, E. Wenger, V. Wulf (Eds.), *Communities and Technologies*, Kluwer, Deventer 2003, pp. 81–96.

Van Alstyne M., Zhang J., EmailNet: A system for automatically mining social networks from organizational email communication, [in:] *NAACSOS Conference 2003 Proceedings*, CASOS, Pittsburgh 2003, http://www.casos.cs.cmu.edu/events/conferences/2003/proceedings/day1.zip.

Wellman B., Computer networks as social networks, *Science* 2001, Vol. 293, pp. 2031–2034.

Yoo Sh., Yang Y., Lin F., Moon I.-C., Mining social networks for personalized email prioritization, In: *Proceedings of the 15th International Conference on Knowledge Discovery and Data Mining*, ACM, Paris 2009, pp. 967–976.

## WSPOMAGANIE ZARZĄDZANIA PROJEKTAMI ZA POŚREDNICTWEM ANALIZY POCZTY ELEKTRONICZNEJ

**Streszczenie:** W artykule przedstawiono i opisano platformę do analizy poczty elektronicznej zorientowanej na wspomaganie zarządzania projektami organizacji zarówno biznesowej, jak i niebiznesowej. Zakłada się pozyskiwanie danych z serwerów IMAP i ich przetwarzanie w celu tworzenia raportów dotyczących czynności komunikacyjnych pomiędzy udziałowcami projektu w formie graficznej i tabelarycznej. Raporty mogą być użyteczne w analizowaniu wewnętrznej komunikacji członków zespołu projektowego, identyfikowanie kluczowych członków zespołu odpowiedzialnych za komunikację z zewnętrznymi partnerami, a także zaobserwowanych jako realizujących powyżej średnich czynności komunikacyjne. Projekt został zrealizowany w środowisku *open-source* Python.

**Słowa kluczowe:** e-mail, zarządzanie projektami, analiza.