EMailAnalyzer: An E-Mail Mining Plug-in for the ProM Framework

Wil M.P. van der Aalst[1] and Andriy Nikolov[2]

[1] Department of Information Systems, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands.
[2] Knowledge Media Institute, The Open University, Milton Keynes, United Kingdom.
w.m.p.v.d.aalst@tue.nl, a.nikolov@open.ac.uk

Abstract. Increasingly information systems log historic information in a systematic way. Workflow management systems, but also ERP, CRM, SCM, and B2B systems often provide a so-called "event log", i.e., a log recording the execution of activities. Thus far, process mining has been focusing on such structured event logs resulting in powerful analysis techniques and tools for discovering process, control, data, organizational, and social structures from event logs. Unfortunately, many work processes are not supported by systems providing structured logs. Instead very basic tools such as a text editors, spreadsheets, and e-mail are used. This report explores the application of process mining to e-mail, i.e., unstructured or semi-structured e-mail messages are converted in event logs suitable for the application of process mining tools. This report presents the tool *EMailAnalyzer* which analyzes and transforms e-mail messages in MS Outlook to a format that can be used by our process mining tools. The main innovative aspect of this work is that our analysis is not restricted to the social network, the main goal is to discover interaction patterns and processes.

Keywords: Process mining, Social network analysis, Computer supported cooperative work, Workflow management.

1 Introduction

Buzzwords such as BAM (Business Activity Monitoring), BOM (Business Operations Management), BPI (Business Process Intelligence) illustrate the interest in closing the BPM loop [1, 2]. This is illustrated by Figure 1 which shows the level of support in four different years using the BPM lifecycle. The lifecycle identifies four different phases: *process design* (i.e., making a workflow schema), *system configuration* (i.e., getting a system to support the designed process), *process enactment* (i.e., the actual execution of the process using the system), and *diagnosis* (i.e., extracting knowledge from the process as it has been executed). As Figure 1 illustrates, BPM technology (e.g., workflow management systems) started with a focus on getting the system to work (i.e., the system configuration phase). Since the early nineties BPM technology matured and more emphasis was put on supporting the process design and process enactment phases in a better way. Now many vendors are trying to close the BPM lifecycle by adding

diagnosis functionality [3]. The buzzwords BAM, BOM, BPI, etc. illustrate these attempts.

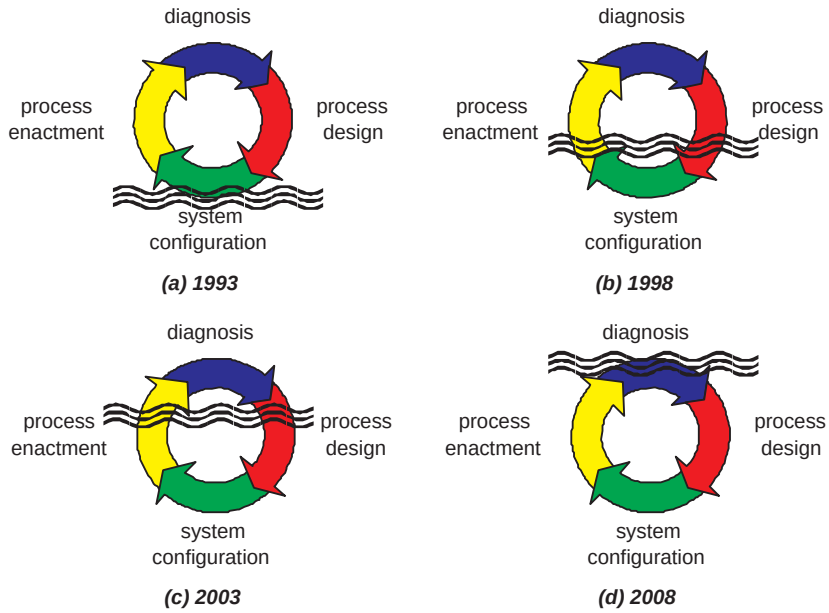

Fig. 1. The level of support is rising: Closing the Business Process Management (BPM) cycle.

The diagnosis phase assumes that data is collected in the enactment phase. Most information systems provide some kind of *event log* (also referred to as transaction log or audit trail). Typically such an event log registers the start and/or completion of activities. Every event refers to a case (i.e., process instance) and an activity (i.e., the step in the process executed), and, in most systems, also a timestamp, a performer, and some additional data. Process mining techniques [3–9] take an event log as a starting point to extract knowledge, e.g., a model of the organization or the process. In the context of our ProM tool [10] we are able to extract different types of process models (e.g., Petri nets, event-driven process chains, and instance graphs), social networks, etc. Moreover, we can check properties (using an LTL-like language) and measure conformance (how well does a model fit with the observed behavior?).

Existing techniques for process mining assume an event log to be in place. For many process-aware information systems [2] this assumption is valid. For example, Workflow Management (WFM) systems, Enterprise Resource Planning (ERP) systems, Customer Relationship Management (CRM), Case Handling (CH) and Product Data Management (PDM) systems log information in some transaction log or audit trail. New legislation such as the Sarbanes-Oxley (SOX)

Act [11] and increased emphasis on corporate governance has triggered the need for improved auditing systems [12]. To audit an organization, business activities need to be monitored. As enterprises become increasingly automated, a tight coupling between auditing systems and the information systems supporting the operational processes becomes more important. However, *many business processes are not directly supported by some process-aware information system.* For many work processes relatively simple tools such as an e-mail program and text editor are being used. E-mail can be seen as the most popular tool used for Computer Supported Cooperative Work (CSCW) [13–15]. The CSCW domain provides a very broad range of systems that support "work" in all its forms. WFM systems and other process-aware information systems can be seen as particular CSCW systems aiming at well-structured office processes. Therefore, it is worthwhile to explore the application of process mining in the broader CSCW domain. In this report, we focus on e-mail systems and their logs. We will show that it is possible to apply process mining techniques to a widely used e-mail program like Microsoft Outlook.

E-mail is widely used for communication inside organizations and between organizations. Analysis of e-mail communication [16–20] is a popular topic of research in social sciences, in particular sociometry [21–30]. If the tasks in the work processes of the organization involve different employees, then they need to communicate to perform the business process. Assigning tasks, asking for more information, reporting results - all these activities are performed by sending e-mail messages. Such e-mail messages necessarily contain process-related information to make recipients understand them. If it is possible to extract the process-related information from such messages, then they may serve as an event log of the organization. The main problem in this case is the extraction of information. A normal e-mail message does not contain any explicit information which allows to make a conclusion about its relations to a particular process instance or task or its relevance to the organization's business process at all. Therefore the e-mail messages must be either tagged with this data in a standard way before sending or this data must be somehow extracted from the message's content and its meta data (e.g., recipients, subject). The first variant (automatic tagging) is applicable if the messages are sent not directly by e-mail client but by the process-aware transaction system. Very often such systems include e-mail sending functionality. In this case it is easy to adjust the functionality of such a system to tag the e-mail with the available information about the current process instance and the task being performed. The second variant (extracting data from arbitrary messages) is more involved. Considering the message topic and text can help to make a conclusion about the link between the message and the underlying workflow. Also the sender or recipient can be actively involved in making this link. This report describes an approach for extracting the process event logs from the e-mail logs and presents a concrete tool: EmailAnalyzer. This tool can extract the information from the e-mail program and transforms into a format useable by ProM, our process mining framework [10].

The report is organized as follows. Section 2 describes a running example, which will be used to illustrate the functionality of the tool. Section 3 describes the main steps, which must be performed by the tool to retrieve a process log from an e-mail log. Section 4 discusses related work. Finally, Section 5 concludes the report and lists possible directions for the future work. For the detailed information about the process mining issues and approaches we refer to [3].

2 Running Example

In this report we will use a running example to discuss our work and present the *EMailAnalyzer* tool. This section introduces the example by describing the process, the organization, and and example log. The example is inspired by a real-life case within a metallurgic enterprise.

2.1 Process

Figure 2 shows a business process within the metallurgic enterprise. The process describes how raw materials are bought. The workflow process is initiated by a business proposal from a potential supplier, who proposes to sell the materials. The task "Proposal from the supplier" represents the discussion of possible conditions between the supplier and a member of the company's supply department. After that the member of the supply department considers the conditions and decides if they are acceptable ("Evaluation" task). The member of the supply department performs this evaluation based on his experience. She or he may consult other people or compare the proposal with other proposals but there is no standard evaluation procedure. Therefore it is unlikely that this stage is reflected in the e-mail log. After evaluation the supply department person decides if the proposal is interesting for the company or not. If its conditions are not acceptable, then the process is cancelled. Otherwise the supply department person contacts a member of the legal department and asks him to prepare the contract. The legal department employee prepares the contract and signs the agreement with the supplier. The next stage of the workflow process is triggered when the negotiated materials are delivered. An employee from the supply department checks the quality of the supplied materials. If the quality (usually chemical structure) completely corresponds to the conditions of the contract, then the employee asks the financial department to transfer the payment to the supplier. If the quality is unsatisfactory, then the supply department person initiates the cancellation of the case. In Figure 2 it is shown as one task "Cancel" for simplicity reasons but in fact it includes several subtasks (returning the materials back to the supplier, preparing corresponding documents etc.). The third variant is when the quality does not satisfy the conditions of the contract but the agreement should not be reversed completely. It is possible when only a part of the consignment is not of satisfactory quality or if the materials are applicable but correspond to the different price category. In this case the legal department

person is contacted to renegotiate the conditions of the contract with the supplier ("Change conditions of the contract"). If renegotiation succeeds, then the contract is signed again and the legal department person contacts the financial department to transfer the payment to the supplier's account. Otherwise the cancellation of the case is triggered.

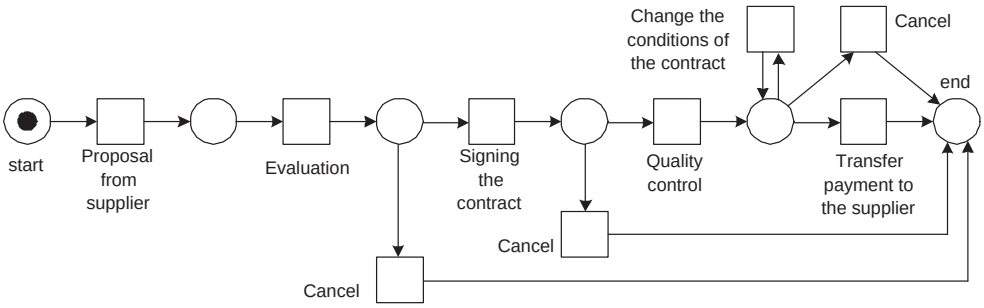

Fig. 2. One of the processes of the metallurgic enterprise.

2.2 Roles

Three subunits of the company are involved in the workflow process: the supply department, the legal department and the financial department. The legal department tends to use case management, i.e., usually the same person from the legal department is involved in different stages of the workflow case. However, preliminary negotiations with the supplier and quality control can be performed by different people from the supply department. In our running example, we assume that the departments consist of the following people:

- Supply department: John, Mike, and Sue.
- Legal department: Carol and Pete.
- Financial department: George and Joan.

2.3 E-mail Log Example

The e-mail log shown in Table 1 represents a set of messages, which reflects processing of six workflow instances, i.e., cases. Each workflow instance is related to a potential supplier company. If the sender or the recipient of a message is from an external company, then company's name is given in brackets. The task "Evaluation" does not appear in the e-mail log because it is usually performed by the same employee as the "Proposal" task.

From	To	Topic
Mark(UkrAlum)	Sue	Business Proposal for New Plant
Joe(MetalGroup)	John	Business proposal
Michael(AluSteel)	Sue	Proposal AluSteel
Sue	Pete	Prepare contract with AluSteel
John	Pete	Change Conditions AluSteel
John	Carol	Contract with MetalGroup
Mike	Carol	Change Conditions with MetalGroup
Ann(CvetMet)	Mike	Business proposal
Mike	Carol	Contract with CvetMet needed
Sue	Carol	Prepare Contract with UkrAlum
John	Joan	Transfer Payment for UkrAlum
Linda(Sarmat)	Mike	Commercial proposal
Mike	Pete	Cancel Sarmat: No interest
Mike	Carol	Update CvetMet Conditions
Carol	Mike	Cancel CvetMet
Carol	Joe	Conditions of Metalgroup
Carol	George	Transfer payment for MetalGroup
Sarah(Metox)	Sue	Business Proposal for New Plant
Sue	Carol	Prepare Contract with Metox
Mike	Pete	Update AluSteel Conditions
Mike	Pete	Yet another change of AluSteel Conditions
Pete	Mike	AluSteel is Cancelled
Carol	Sue	Cancel Metox

Table 1. A set of messages to handle the processing of the case

3 The Mining Process

The goal of the mining process is to produce the process log in the standard XML format supported by the ProM tool [10] from the set of messages stored in the users' inbox folders. The process consists of several steps. The first step is to extract the e-mail logs in the XML format from the user's Outlook e-mail clients. The next step is to pre-process the e-mail log in order to remove ambiguities in recipient's names and to exclude irrelevant e-mail messages from the future analysis. The e-mail log can be used to perform social network analysis and to generate the process log. The process log resulting from such an analysis can serve as input for wide variety of process mining tools, e.g., the many mining plug-ins present in ProM.

3.1 Extracting E-mail Logs

The first step is to extract the e-mail log of an organization from the e-mail clients of all users. A separate tool - *InboxLoader* is used for this purpose. This tool uses COM interface of the Microsoft Outlook to retrieve the list of messages from the user's inbox folder. Afterwards InboxLoader saves this list of e-mail messages

into an XML-file of the standard format recognized by the *EmailAnalyzer* tool. After retrieving e-mail logs of each user all these message logs are loaded into the EmailAnalyzer tool, which combines them together. Note that both the InboxLoader and EmailAnalyzer are embedded in the ProM framework.

3.2 Pre-processing Stage

Very often the same person can appear in the recipient lists of different e-mail messages under different names. For example, inbox folder, which was used for tests, contains messages addressed to "`anikolov`", "`anikolov@cs.vu.nl`", "`'anikolov'`", "`anikolov@few.vu.nl`", "`Andriy Nikolov`". All these names in fact are related to only one mail account. This may happen because of the following reasons:

- Different e-mail clients may fill recipient names in different ways depending on their settings and destination of the message (inside the network domain or outside).
- The same mail account may be referred in different ways depending on the settings of the server (e.g. "`anikolov@cs.vu.nl`" and "`anikolov@few.vu.nl`" are in fact the same mail address).
- One person may use several mail accounts.

The goal of the consolidation stage is to identify all participants of the e-mail exchange and group together all messages related to the same (physical) recipient. It is performed by finding all names referring to the same person. This process can be automated by comparing names and e-mail addresses (and parts of e-mail addresses). The goal of the next step is to select from the whole set of messages only those, which are needed for analysis. This step is necessary because usually the mailbox of a person contains many messages irrelevant for the process or organization. These may include private messages, messages to and from the people outside an organization or simply spam messages. Considering these messages during the analysis stage may considerably distort the results of the analysis. Therefore these messages must be excluded. Criteria of relevance depend on the goals of analysis, which are determined by the user, so this step cannot be performed automatically and the user has to take care of it. The user can either exclude irrelevant messages one by one or exclude/include recipients together with all messages linked to them. Resulting message list can afterwards be used for analysis.

3.3 Mining Social Networks: Sociograms and Messages Frequency Charts

Sociograms: A Static View Based on the set of e-mail messages from all members of the organization, it is possible to build a sociogram in the form of a directed graph. Nodes of the graph will represent particular users and the arcs indicate communication between them. All arcs have weights, which describe the

frequency of messages. Each e-mail message from user A to user B will increase the weight of the arc from A to B by 1. A special case is a message with several recipients. The recipients may belong to different categories. Usually a message is addressed to the recipients listed in the "To" category while "Cc" and "Bcc" recipients are only supposed to be informed of it. Because of this it can make sense to assign different weights to the recipients of different categories. For example, a message from the person A to a person B with a carbon copy sent to the person C should increase the arc from A to B by 1 and the arc from A to C by 0,5. After processing all messages the weights in the network can be normalized by dividing them by the maximal weight in the network, i.e., $w_{i,j}^{norm} = w_{i,j}/w_{max}$.

A sociogram built after the exclusion of all reply messages may also appear interesting. Very often during the execution of a business process employees may have long e-mail conversations while performing one task together or two tasks in order. For example, after person A gave a task to person B, person B may reply with a request for some additional information from person A and receive this information in return. In another similar case person B may not need such a request. It does not mean that in the first case relations between two people is stronger. To avoid counting of such auxiliary messages it may be helpful to exclude all reply messages from consideration. The resulting sociogram matrix can be saved in the Agna and NetMiner data file format (like in MiSoN tool [31]) where advanced analysis methods can be applied to it.

Message Frequency Charts: A Dynamic View Building a sociogram based on the list of e-mail messages may lead to the partial loss of information. A sociogram represents only the static view on the information exchange inside an organization. However the log of e-mail messages also contains information about the dynamics of this information exchange process. Each e-mail message has a timestamp, which shows when it was sent. The user may want to see how the frequency of e-mail messages inside an organization changed during some period of time. This information can be used in different ways. For example, it can help the manager to make decisions about the schedule of projects (e.g. if (s)he sees that in the past periods of intense message flows alternated with the periods of less e-mail activities then (s)he may decide to revise the planning of work in future).

A dynamic view of the information exchange can be shown with the help of a chart, showing frequency of the messages. The period of time, which the user wants to consider, is divided into the set of intervals of fixed length. The messages in the message log are mapped on these intervals based on the time/date of sending. X-axis of the chart represents the time while the Y-axis shows the number of messages, which were sent during each interval of time.

Sociogram Example After loading the e-mail log we should add for each of the external persons one additional alias - the name of his company and make it his main name. Before building a sociogram we should exclude these external users

from analysis if we are interested only in communication between the employees. The sociogram built on the base of the e-mail log is shown in Figure 3.

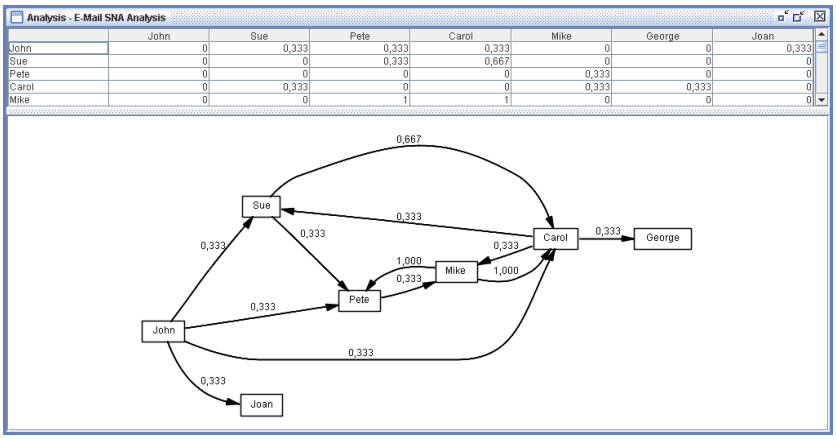

Fig. 3. A sociogram built on the basis of an e-mail log.

3.4 Mining Process Logs

Translation of e-mail logs into process logs by the EMailAnalyzer tool is based on the tags in the message's subject, which describe the relation between the message and process case and task. The tags represent the names of the case and task. There are three possible situations:

- The tags are added automatically by a corporative process-aware system in some standard way;
- The user (either the recipient or the sender) is forced (or stimulated) to classify each e-mail message, i.e., the user explicitly tags the message;
- The tags are not added explicitly and the text in the subject field is only intended for the recipient's understanding.

In the first two cases case there are explicit tags (added by the system or by users), which makes the mining procedure trivial. The latter situation means that there is no tagging standard and that the mining algorithm must figure out the relations between each message and process cases/tasks. Although we consider the first two scenarios more realistic, we will also show how our tool can deal with the latter situation.

Tags in E-mail Messages Automatic annotation of the e-mail messages by the corporative system can be useful under following conditions:

– E-mail exchange functionality of the system is used to send messages, which are relevant to the business process. For example, many groupware systems use e-mail as a transport layer.
– The system operates with such process-related notions as "case", "task" and "event".
– The system does not contain explicit process description (otherwise mining is less useful since it can only be used for conformance testing and Delta analysis).
– The system does not generate process log directly (otherwise it is easier to mine the process description from the process log).

The current version of the EmailAnalyzer tool assumes that the messages related to the business process should be tagged in following way:

– The messages are linked to process cases: the case is determined by the contact person linked to the message (on View/Options tab). This person represents a customer linked to the case.
– The messages are linked to tasks: the task name and phase are explicitly included into the subject in brackets "`{}`" and divided by the "`^`" symbol. Phases of each task are the same as event types in the log definition (from "schedule" to "complete").

If we restrict our mail clients to MS Outlook there is another option: MS Outlook allows adding user-defined fields to the messages. In this case tags can be added as such user-defined fields, which should not be visible in the subject line. Such a solution is probably more convenient.

E-mail Analyzer Process Log Building Functionality As indicated before, the tags added to messages can be useful only in case when the organization uses a special groupware system, which should annotate the messages with all necessary process-related information automatically. If there is no such system and the users have to edit and send messages manually, then it is inconvenient for them to write all the message tags according to the strict rules described above. Parts of process-related information can be contained in any part of the message description. Therefore it was decided to add to the functionality of EMailAnalyzer the possibility to choose among different settings. These settings will determine how process-related information should be extracted from the e-mail log. Each message represents one audit trail entry in the process log. The process related information includes the description of (1) the *case*, (2) the *task* and (3) the *event* related to the message. On the process log options screen of EMailAnalyzer, the user can specify various options regarding each of these three categories.

First of all, the user can specify how to extract the *case* from the log. Several options are possible:

– **Linked contact** (default):
The default option means that the case describes the customer and is specified by the contact person linked to the message.

– **Message sender**:
  This option means that the case is represented by a person who sent the message.
– **Message recipient**:
  This option specifies that the case, to which the message belongs, corresponds to one of the message recipients.
– **Subject**:
  The subject line is used as the description of the case. If only a part of the subject is used as a case description then the user can check the flag "Partial coincidence allowed".
– **Any**:
  Sometimes the case is represented by the customer, which is referred in the inter-organizational communication in the subject line. The option "Any" means that the case name should be searched both in the sender/recipients list and the subject line.
– **Single case**:
  Selecting this option assumes that all messages in the log are related to one single case.
– **Thread**:
  Thread of messages represents an initial message together with all replies to it. This option assumes that the case is described by the subject line of the message and includes all messages with this subject together with all replies.

After selecting one of the options listed, the user can see the list of possible case names. The user can then edit the list by adding/deleting new case identifiers thus effectively filtering the log.

Second, there are options to derive the identity of the task. It is assumed that the task name is included into the message subject. It can be stored in a standardized or arbitrary way. The user has two options:

– **Default** (i.e., between `{` and `}`):
  The task name is included into the subject line according to the standard format described in the previous section (in the subject line between "`{`" and "`}`" brackets together with the event type);
– **Anywhere in the subject**:
  There is no standard format of the subject line. The task name can be either the whole subject line or its part. As for the cases, this is determined by the "Partial coincidence allowed" flag.

Similar to cases, the user can edit the list of possible task names manually.

Third, there are options to derive the event type. By default EMailAnalyzer considers seven event types used by the ProM framework and defined in the MXML format [3, 10]: "schedule", "start", "complete", "suspend", "resume", "withdraw" and "abort". The name of the event type should be mentioned in the message subject either according to a standardized syntax described above or in any place of the subject line. It is possible that the organization considers another set of event types (e.g. "start" and "finish"). In this case the set of event

types used by the organization must be mapped into the standard set of event types described in [3]. EMailAnalyzer allows performing the mapping using the alias names of the standard event types. Alias name represents the name, under which the event type will appear in the e-mail log. The user can change this alias name. For example, (s)he can change the name of the "complete" event type to "finish". After that if the tool will find the word "finish" in the message subject it will create an audit trail entry with the event type "complete". The user can also add his own event types. They will have the standard name "unknown". The user has following options considering the source of the event type description in the message:

- **Default** (i.e., between `{` and `}`):
  As for the tasks, this means that event type name is stored according to the standard described above (in the message subject).
- **Anywhere in the subject**:
  Event type can appear in any place in the subject line
- **One event type** (i.e., "normal"):
  In this case each message represents an audit trail entry with the type "unknown". The attribute "unknowntype" will be set to "normal".

Checkbox "Try to insert missing events" is applicable only if the standard set of events is used. In this case the system tries to insert audit trail entries to the process log so that each task in the process log will have complete set of corresponding audit trail entries. Completeness in our case means that each task has corresponding set of events, which lead from the state "new" to one of the terminal states "completed" or "terminated" [31].

After all the options determining the mining process are specified, the main translation algorithm to translate the e-mail log into the process log can be executed. The algorithm is as follows:

- Find all tagged messages in the e-mail log according to the user's settings.
- Distribute messages between cases.
- For each case do:
  - sort messages by the delivery date;
  - create for each message one AuditTrailEntry;
  - save the case and its AuditTrailEntry records.

The algorithm for adding missing events is activated after the main process log building algorithm. Its steps are as follows:

- Generate the process log from the e-mail log.
- For each task do:
  - select all events with the standard types (not "unknown");
  - sort the events by time;
  - for each event check the following possibilities. If the event is not an initial event ("schedule") and is not a direct successor of the previous event (e.g. "complete" is a direct successor of "start" but not "schedule" or "suspend") then add to the log a new event with following properties.

The type should be equal to the predecessor of current event type. The timestamp should match the current event time minus 1 minute (note that this is an arbitrary time). If the event is not a final event ("abort", "terminate" or "complete") and is not a direct predecessor of the next event then add to the log a new event with following properties. The type should be equal to the successor of current event type. The timestamp should match the current event time plus 1 minute (again we choose an arbitrary time).

By default the "successful" path ("scheduled - start - complete") is considered as standard. It means that, if possible, the events from this path are selected as direct predecessors/successors. For example, the event "complete" is considered as a default successor for the events "resume" (and not the events "abort" or "suspend", which are also possible in the state "active").

Process Log Building Example Now it is time to return to the e-mail messages shown in Table 1 and try to derive a process model using ProM. To build the process log from our example e-mail log we have to select following options on the "Process log setup" page:

- The case source is "Any field" and the "Partial coincidence allowed" flag is set. The list of case names consists of the names of the companies. This way cases are detected by the company involved.
- The task source is "Anywhere in the subject" and the "Partial coincidence allowed" flag is set. The list of keywords for task names consists of "proposal", "contract", "condition", "cancel" and "payment". Figure 4 shows how to set these parameters.
- The event source is "One event type", i.e., there are no different event types distinguished.

Figure 5 shows the process model that we are trying to discover. As indicated before, tasks "Evaluation" and "Quality control" are assumed to be invisible, i.e., these tasks occur but may not result in the sending of e-mail messages. The other tasks are identified by the keywords "proposal", "contract", "condition", "cancel" and "payment" as indicated in Figure 5.

Using these settings it is possible to generate a process log. Since EMailAnalyzer is embedded in ProM the resulting process log can be stored but also analyzed using one of the many mining algorithms present in ProM. In the remainder of this section, we show of the results we obtained by applying the various plug-ins in ProM.

The α-algorithm [4] was one of the first process mining techniques able to deal with processes exhibiting concurrency. ProM provides a plug-in for this algorithm and if we apply it to the log generated by EMailAnalyzer we obtain the model shown in Figure 6. The α-algorithm produces a Petri net reflecting only the control-flow in the underlying process. Note that the tasks "Evaluation" and "Quality control" are missing in the Petri net. This makes sense because

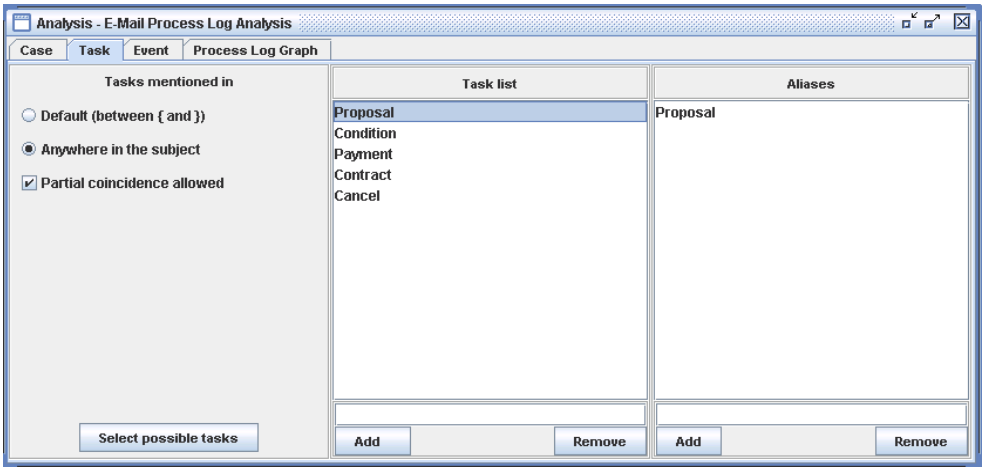

Fig. 4. A screenshot of the interface used to select the options when extracting processes from e-mail logs.

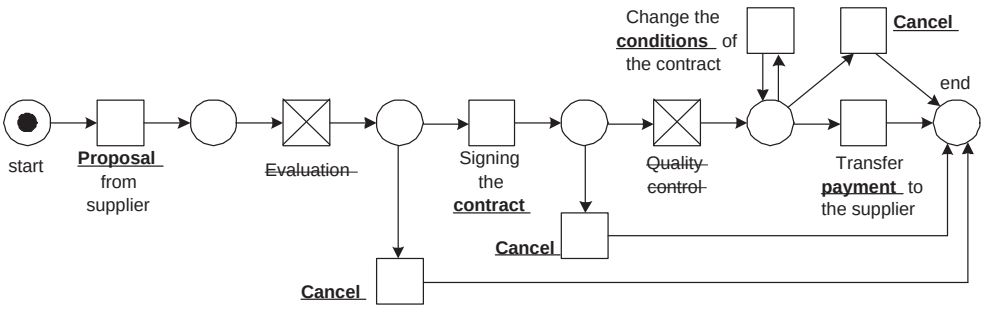

Fig. 5. The tasks "Evaluation" and "Quality control" are invisible and the other tasks are characterized by the keyword indicated.

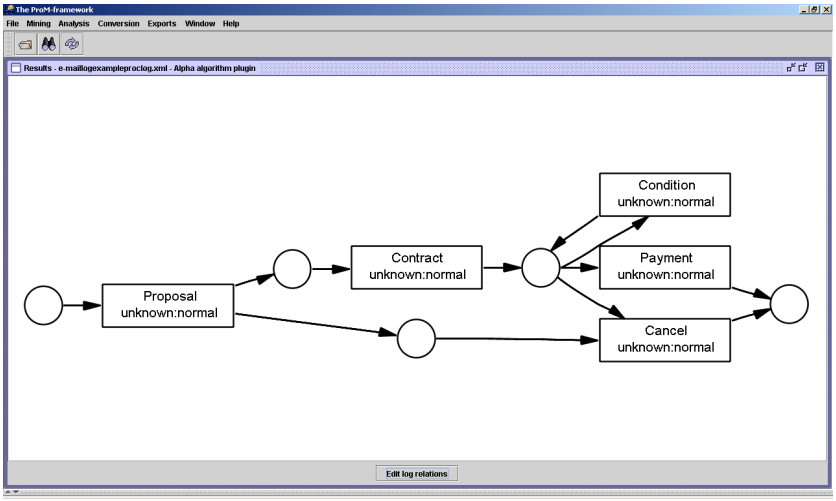

Fig. 6. A process model built on the basis of an e-mail log using the α plug-in.

these are not recorded. Moreover, instead of three cancellation tasks there is just a single "Cancel". This also makes sense since EMailAnalyzer looks for the keyword "cancel". Hence it cannot distinguish between the three instances of "cancel" shown in Figure 5. The α-algorithm has problems dealing with duplicate tasks and as a result the model shown in Figure 6 is not completely accurate. Fortunately, ProM offers more powerful mining techniques as shown below.

The multi-phase mining plug-in of ProM is based on so-called instance graphs [32]. In contrast to the α-algorithm it does not try to discover the process model in a single step. Instead, it first builds a model (i.e., an instance graph) for every case and only then aggregates these instance graph into an aggregated instance graph. The result can be visualized in terms of an Event-driven Process Chain (EPC) or a Petri net. Figure 7 shows the result in terms of an EPC. The left window shows the entire process while the right window zooms in on the interesting part. An EPC consists of three types of nodes:

Functions
: The basic building blocks are functions. A function corresponds to an activity (task, process step) which needs to be executed.

Events
: Events describe the situation before and/or after a function is executed. Functions are linked by events.

Connectors
: Connectors can be used to connect functions and events. This way, the flow of control is specified. There are three types of connectors: $\wedge$ (and), $\times$ (xor) and $\vee$ (or).

In Figure 7 all connectors are of type $\times$, i.e., an XOR-join or an XOR-split. A close observation shows that Figure 7 indeed captures the original process model

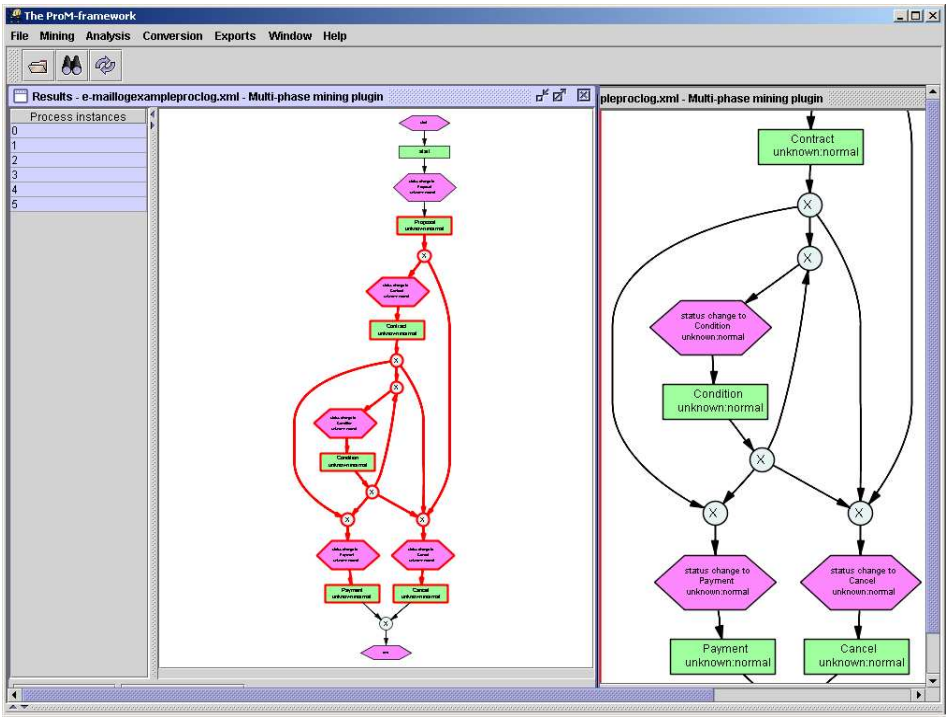

Fig. 7. A process model built using the multi-phase plug-in.

after abstracting from "Evaluation" and "Quality control", and joining the three cancellation tasks into one. Figure 7 shows that a case can be cancelled at three points in the process. A payment follows after the initial contracting step of after changing the conditions.

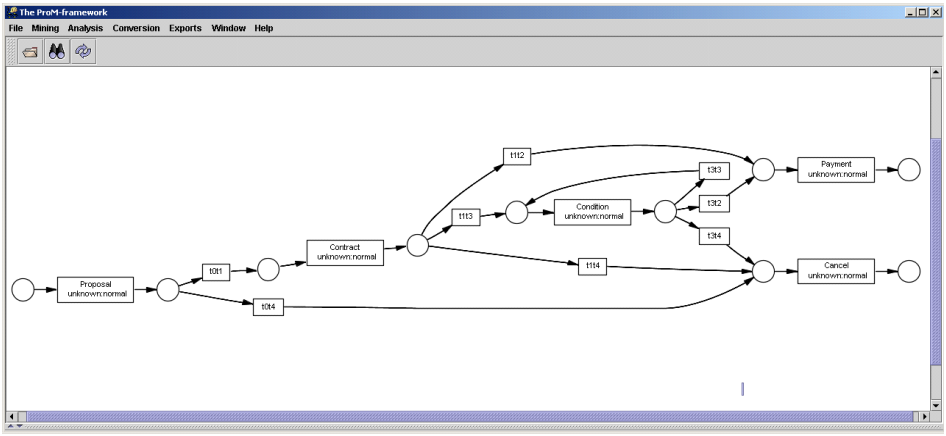

Fig. 8. A process model built using the genetic miner plug-in.

As will be discussed in more detail in Section 3.6, logs contain different types of noise. The fact that for one case a certain path was followed does not imply that this should be considered as part of the process model. To tackle these problems, ProM offers alternative process mining methods. For example, the genetic miner plug-in of ProM uses a genetic algorithm to discover process models. Figure 8 shows the result in terms of a Petri net. The model is behavioral equivalent to the original process model after abstracting from "Evaluation" and "Quality control", and joining the three cancellation tasks into one. If we compare Figure 5 and Figure 8 we see that the invisible tasks have been removed but that "routing tasks" have been added by the algorithm to reflect choices. For example, after executing task "Condition" there are three transitions enabled: "t3t3" to again renegotiate the contract, "t3t2" to do the payment, and "t3t2" to cancel after renegotiation. This way the genetic algorithm avoids the problem the α-algorithm could not address (cf. Figure 6).

Figures 6, 7, and 8 shows only a small part of the full functionality of the ProM framework. The three plug-ins shown are part of a set of more than 30 plug-ins involving mining algorithms, analysis routines, conversion functions, social network analyzers, conformance checkers, and LTL checkers. Just to illustrate a bit of this functionality, we show two analysis plug-ins and the LTL-checker plug-in.

Figures 9 shows two types analysis. The left-hand part of the window shows the result of applying Petri-net based analysis techniques to the model generated by the α-algorithm (i.e., Figure 6). The coverability graph shows that a token

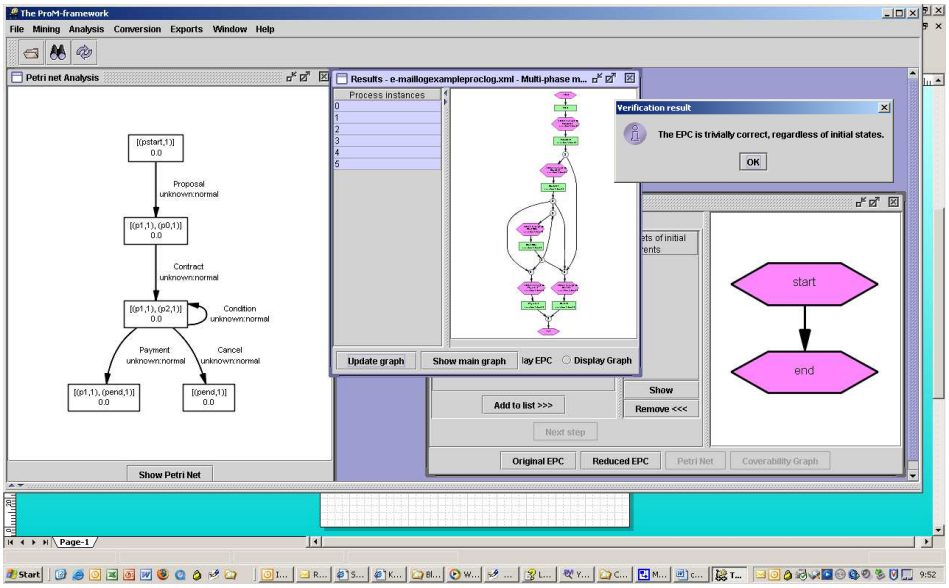

Fig. 9. The Petri net analysis plug-in is able to find the error in Figure 6 while the EPC in Figure 7 can be shown to be correct by reduction.

may get stuck if task "payment" is executed. The right-hand part of the window shows the result of analyzing the EPC in Figure 7 (i.e., the result of multi-phase mining). As shown, the EPC can be reduced to the empty process, thus demonstrating that it is *sound* [33].

Deviations from the "normal process" may be desirable but may also point to inefficiencies or even fraud. New legislation such as the Sarbanes-Oxley (SOX) Act [11] and increased emphasis on corporate governance has triggered the need for improved auditing systems [12]. To support this we have developed the LTL checker plug-in for ProM [34]. Figure 10 shows a screenshot while checking the *4-eyes principle*. This principle says that although authorized to execute two activities, a person is not allowed to execute both activities for the same case. For example, a manager may submit a request (e.g., to purchase equipment, to make a trip, or to work overtime) and (s)he may also approve requests. However, it may be desirable to apply the 4-eyes principle implying that the manager is not allowed to approve his own request. If there is an event log recoding the events "submit request" and "approve request", the 4-eyes principle can be verified easily. If we try to apply the 4-eyes principle to the running example, we could argue that the same person is not allowed to handle "Contract" and "Conditions". The left-hand part of the window shown in Figure 10 shows the form where one can select a desired or undesired property and set its parameters. The right-hand part of the window shows the result. The property does not hold for two of the six cases in our example. As shown each of the cases can be inspected further. Note that the LTL checker is based on *Linear Temporal Logic* (LTL)

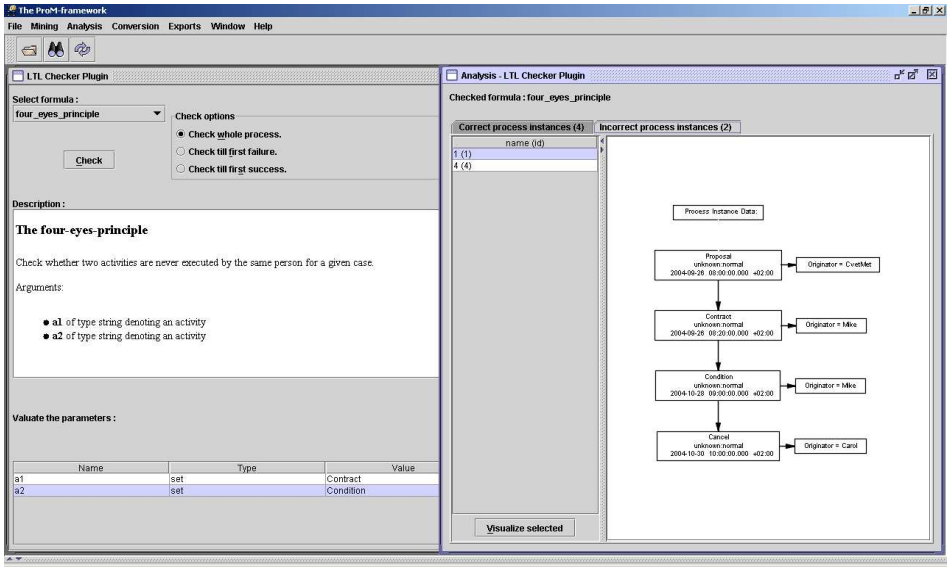

Fig. 10. The LTL-checker plug-in is able to check for various properties, e.g., detection of fraud.

[35, 36] and tailored towards event logs holding information on activities, cases (i.e., process instances), timestamps, originators (the person or resource executing the activity), and related data. Hence it can be used to specify properties much more advanced than the 4-eyes principle.

3.5 User Involvement Diagrams

When considering information about existing business process it can be useful to study the links between each user and the tasks and cases in which (s)he is involved. These links may help to make conclusions about the workload of each user and to redistribute the functions between employees if necessary. EmailAnalyzer offers two *user involvement diagrams* to visualize these links. The *task-person diagram* shows the relations between tasks and users. If a user executes a given task frequently, there is a strong link between both. Similarly the *case-person diagram* displays the relationships between cases and users.

Recall that EMailAnalyzer translates the e-mail log into the process log assuming that one e-mail message corresponds to a single event. Several events may refer to the same task. We assume that all senders and recipients of all messages related to a task are in some way involved in this task. The task-person diagram shows the links between each employee and the tasks, in which (s)he was involved. The diagram represents a graph, which has nodes of two types: employees and tasks. Each arc connects one person with one task. The weight of each arc is proportional to the number of times a certain employee was involved in a certain task. All weights are normalized by dividing them by the

maximal weight in the whole graph. When displaying the diagram the user has an option to display for each employee only the arcs with the maximal weight. This may help to make the diagram more readable by showing for each employee only the tasks, in which (s)he was most involved.

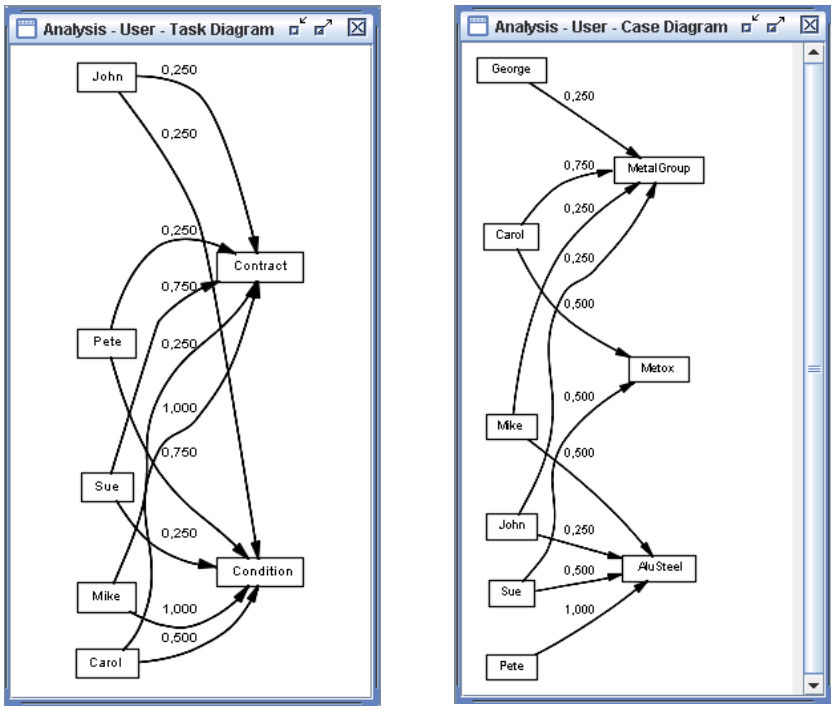

Fig. 11. Two user involvement diagrams: the task-person diagram (left) and the case-person diagram (right).

The task-person diagram based on the process log of the running example is shown in Figure 11 (left). Note that it is possible to see which tasks involve external communication and which are performed completely inside the company. With regard to the employee's occupation we can distinguish the cluster linked to the "Payment" task only. This allows us to make conclusions about different roles of the performers of these tasks.

A similar diagram can be built for the cases, as shown in Figure 11 (right). The diagram shows how each employee was involved in each process instance reflected in the e-mail log. Arcs in this diagram show the links between each employee and cases. Like in the previous case the weights are proportional to the number of messages related to the case and the employee simultaneously.

3.6 Issues when Mining the Process from the Process Log

Several algorithms have been proposed for process mining. Many of these algorithms cannot deal with concurrency. (The α-algorithm [4] used to construct Figure 6 was among the first to allow for the mining of concurrent processes.) Moreover, existing approaches for mining the process perspective [3–9] have problems dealing with issues such as duplicate activities, hidden activities, non-free-choice constructs, noise, and incompleteness. The problem with *duplicate activities* occurs when the same activity can occur at multiple places in the process. This is a problem because it is no longer clear to which activity some event refers. The problem with *hidden activities* is that essential routing decisions are not logged but impact the routing of cases. *Non-free-choice* constructs are problematic because it is not possible to separate choice from synchronization. We consider two sources of *noise*: (1) incorrectly logged events (i.e., the log does not reflect reality) or (2) exceptions (i.e., sequences of events corresponding to "abnormal behavior"). Clearly noise is difficult to handle. The problem of *incompleteness* is that for many processes it is not realistic to assume that all possible behavior is contained in the log. For processes with many alternative routes and parallelism, the number of possible event traces is typically exponential in the number of activities, e.g., a process with 10 binary choices in a sequence will have $2^{10}(= 1024)$ possible event sequences and a process with 10 activities in parallel will have even $10!(= 3628800)$ possible event sequences.

Real-life logs contain noise (e.g., exceptions or incorrectly logged events) and are typically incomplete (i.e., the event logs contain only a fragment of all possible behaviors). This is highly relevant for the work presented in this report because we can expect e-mail logs to be noisy and incomplete. Therefore, we developed two ways to address the problem: (1) *heuristics* and (2) *genetic algorithms*. The heuristic approach does not only consider binary relations based on direct succession. It allows for thresholds which may incorporate a variety of information, e.g., a task not being the initial or final one should have at least one input and one output. See [9] for more details. Genetic algorithms for process mining use a global search strategy; because the quality or fitness of a candidate model is calculated by comparing the process model with all traces in the event log the search process takes place at a global level. These algorithms start with an initial population of individuals (in this case process models). Populations evolve by selecting the fittest individuals and generating new individuals using genetic operators such as *crossover* (combining parts of two of more individuals) and *mutation* (random modification of an individual). See [37] for more information on our genetic mining algorithm implemented as a plug-in in ProM.

The discussion on current process mining techniques shows that the mining of e-mail messages can benefit from state-of-the-art approaches for dealing with noise, incompleteness and other issues.

4 Related Work

The idea of applying process mining in the context of workflow management was first introduced in [5]. Cook and Wolf have investigated similar issues in the context of software engineering processes using different approaches [6]. It is impossible to point out the many process mining algorithms proposed in literature. However, we would like to mention the α-algorithm [4] which served as a starting point for the ProM framework. For more information on process mining we refer to a special issue of Computers in Industry on process mining [38] and a survey paper [3].

Within the CSCW domain there has been a constant struggle between technological views and sociological views. A nice illustration is the so-called "Winograd-Suchman debate" in the early nineties [39–42]. Winograd and Flores advocated the use of a system called the "coordinator", a system based on Speech act theory (i.e., the language/action perspective) in-between e-mail and workflow technology [39, 42]. People like Suchman and others argued that such systems are undesirable as they "carry an agenda of discipline and control over an organization's members" [41]. Clearly, process mining adds another dimension to this discussion. The goal of process mining is not to control people. However, it can be used to monitor and analyze the behavior of people and organizations. Clearly, such technology triggers ethical questions. However, we consider such questions are beyond the scope of this report. Instead, we focused on the applicability of process mining and Social Network Analysis (SNA) in the presence of e-mail-based logs.

Since the early work of Moreno [27], sociometry, and SNA in particular, have been active research domains. There is a vast amount of textbooks, research papers, and tools available in this domain [21–30]. There have been many studies analyzing organizational activity based on insights from social network analysis. However, some of these studies typically have an ad-hoc character and sociograms are typically constructed based on questionnaires rather than using a structured and automated approach as described in this report. More structured approaches are often based on the analysis of e-mail interaction and additional electronic sources. Several studies have generated sociograms from email logs in organization [16–20] to analyze the communication structure. Such studies have resulted in the identification of relevant, recurrent aspects of interaction in organizational contexts [43, 18]. However, these studies are unable to relate the derived social networks to a particular workflow process, as the analyzed data does not reveal to what activity or case it applies.

Most tools in the SNA domain take sociograms as input. ProM offers two plug-ins that generate sociograms as output: the tool reported in this report and MinSoN [31] a tool to extract sociograms from event logs rather than e-mails.

The tools most related to the work reported in this report are the tools used to discover social networks from the e-mail traffic:

- BuddyGraph (http://www.buddygraph.com/),
- MetaSight (http://www.metasight.co.uk),

– EMailNet (http://emailcommunity.net/).

EMailNet tool is a nice example of a tool which performs social networks analysis based on the e-mail logs. While the EMailNet tool focuses on the social aspects of e-mail communication and *not* on deriving process information, it implements several ideas that can be valuable for the future development of ProM in general and EMailAnalyzer in particular, e.g., coding messages to ensure privacy and retrieving of the e-mail logs from the e-mail server instead of the e-mail clients.

5 Conclusions

This report presented a tool, named EMailAnalyzer, to mine process logs from e-mail logs. The e-mail log is extracted from the user's inbox folders and translated into the process log according to the settings specified by the user. The resulting process log is saved in a standard format, which can be handled by a variety of process mining tools such as the ones implemented in the context of the ProM framework. To illustrate the tool and the ideas behind it, we used a small example to demonstrate that it is really possible to discover process models from e-mail messages. In the example, we assumed by explicit tags. We realize that with the current tool such a scenario is not realistic. From a practical point of view, we consider the explicit tagging of messages vital. Many systems generate e-mail messages than can easily provide tags. Moreover, users (both senders and recipients) can be asked to tag messages or confirm automatically generated tags.[1]

This report and the current version of EmailAnalyzer should be considered as a first step towards fully-automated process discovery from e-mail messages. We plan to extend the functionality of the tool in various directions. For example, a more convenient way of implementing more advanced functionality may be to build a system that directly cooperates with the e-mail server (e.g. Microsoft Exchange). The body of the message should be included into the analysis. In order to cope with the privacy problem, which arrives in this case, a coding mechanism should be implemented so that the content of analyzed messages can be accessed only by the mining algorithm and not by human users. Currently the mining mechanism implements simple keyword-based search to find the links between business process cases and tasks and e-mail messages. The mechanism itself should be improved by adding text mining techniques and natural language processing heuristics to increase the quality of the mining.

Acknowledgement
The authors would like to thank Boudewijn van Dongen for assisting in embedding the EMailAnalyzer plug-in in the ProM framework. We would also like to

[1] It is interesting to note that the "coordinator" system of Winograd and Flores in the eighties already used explicit tagging of e-mail messages [39, 42]. However, these tags were based on Speech act theory (i.e., the language/action perspective) rather than process-related events. Moreover, they did not propose to use this for process mining.

thank him and the rest of the "process mining team", in particular Ton Weijters, Boudewijn van Dongen, Minseok Song, Eric Verbeek, Anne Rozinat, Christian Gnter, and Peter van den Brand, for their on-going work on process mining techniques.

References

1. W.M.P. van der Aalst and K.M. van Hee. *Workflow Management: Models, Methods, and Systems*. MIT press, Cambridge, MA, 2002.
2. M. Dumas, W.M.P. van der Aalst, and A.H.M. ter Hofstede. *Process-Aware Information Systems: Bridging People and Software through Process Technology*. Wiley & Sons, 2005.
3. W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering*, 47(2):237–267, 2003.
4. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
5. R. Agrawal, D. Gunopulos, and F. Leymann. Mining Process Models from Workflow Logs. In *Sixth International Conference on Extending Database Technology*, pages 469–483, 1998.
6. J.E. Cook and A.L. Wolf. Discovering Models of Software Processes from Event-Based Data. *ACM Transactions on Software Engineering and Methodology*, 7(3):215–249, 1998.
7. J. Herbst. A Machine Learning Approach to Workflow Management. In *Proceedings 11th European Conference on Machine Learning*, volume 1810 of *Lecture Notes in Computer Science*, pages 183–194. Springer-Verlag, Berlin, 2000.
8. A.K.A. de Medeiros, W.M.P. van der Aalst, and A.J.M.M. Weijters. Workflow Mining: Current Status and Future Directions. In R. Meersman, Z. Tari, and D.C. Schmidt, editors, *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*, volume 2888 of *Lecture Notes in Computer Science*, pages 389–406. Springer-Verlag, Berlin, 2003.
9. A.J.M.M. Weijters and W.M.P. van der Aalst. Rediscovering Workflow Models from Event-Based Data using Little Thumb. *Integrated Computer-Aided Engineering*, 10(2):151–162, 2003.
10. B.F. van Dongen, A.K. Alves de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, and W.M.P. van der Aalst. The ProM framework: A New Era in Process Mining Tool Support. In G. Ciardo and P. Darondeau, editors, *Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454. Springer-Verlag, Berlin, 2005.
11. P. Sarbanes and G. Oxley et. al. Sarbanes-Oxley Act of 2002, 2002.
12. T. Hoffman. Sarbanes-Oxley Sparks Forensics Apps Interest: Vendors Offer Monitoring Tools to Help Identify Incidents of Financial Fraud. *ComputerWorld*, 38:14–14, 2004.
13. C.A. Ellis. An Evaluation Framework for Collaborative Systems. Technical Report, CU-CS-901-00, University of Colorado, Department of Computer Science, Boulder, USA, 2000.
14. C.A. Ellis, S.J. Gibbs, and G. Rein. Groupware: Some issues and experiences. *Communications of the ACM*, 34(1):38–58, 1991.

15. C.A. Ellis and G. Nutt. Workflow: The Process Spectrum. In A. Sheth, editor, *Proceedings of the NSF Workshop on Workflow and Process Automation in Information Systems*, pages 140–145, Athens, Georgia, May 1996.
16. S. Farnham, S.U. Kelly, W. Portnoy, and J.L.K. Schwartz. Wallop: Designing Social Software for Co-Located Social Networks. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*. IEEE Computer Society Press, Los Alamitos, California, 2004.
17. S. Farnham, W. Portnoy, and A. Turski. Using Email Mailing Lists to Approximate and Explore Corporate Social Networks. In D.W. McDonald, S. Farnham, and D. Fisher, editors, *Proceedings of the CSCW'04 Workshop on Social Networks*, 2004.
18. D. Fisher and P. Dourish. Social and Temporal Structures in Everyday Collaboration. In E. Dykstra-Erickson and M. Tscheligi, editors, *Proceedings of the 2004 Conference on Human Factors in Computing Systems (CHI2004)*, pages 551–558, New York, NY, USA, 2004. ACM Press.
19. B.A. Nardi, S. Whittaker, E. Isaacs, M. Creech, J. Johnson, and J. Hainsworth. Integrating Communication and Information Through ContactMap. *Communications of the ACM*, 45(2):89–95, 2002.
20. H. Ogata, Y. Yano, N. Furugori, and Q. Jin. Computer Supported Social Networking For Augmenting Cooperation. *Computer Supported Cooperative Work*, 10(2):189–209, 2001.
21. A.A. Bavelas. A Mathematical Model for Group Structures. *Human Organization*, 7:16–30, 1948.
22. H.R. Bernard, P.D. Killworth, C. McCarty, G.A. Shelley, and S. Robinson. Comparing Four Different Methods for Measuring Personal Social Networks. *Social Networks*, 12:179–216, 1990.
23. R.S. Burt and M. Minor. *Applied Network Analysis: A Methodological Introduction*. Sage, Newbury Park CA, 1983.
24. M. Feldman. Electronic Mail and Weak Ties in Organizations. *Office: Technology and People*, 3:83–101, 1987.
25. L.C. Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40:35–41, 1977.
26. L.C. Freeman. Centrality in Social Networks: Conceptual Clarification. *Social Networks*, 1:215–239, 1979.
27. J.L. Moreno. *Who Shall Survive?* Nervous and Mental Disease Publishing Company, Washington, DC, 1934.
28. H. Nemati and C.D. Barko. *Organizational Data Mining: Leveraging Enterprise Data Resources for Optimal Performance*. Idea Group Publishing, Hershey, PA, USA, 2003.
29. J. Scott. *Social Network Analysis*. Sage, Newbury Park CA, 1992.
30. S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, Cambridge, 1994.
31. W.M.P. van der Aalst and M. Song. Mining Social Networks: Uncovering Interaction Patterns in Business Processes. In J. Desel, B. Pernici, and M. Weske, editors, *International Conference on Business Process Management (BPM 2004)*, volume 3080 of *Lecture Notes in Computer Science*, pages 244–260. Springer-Verlag, Berlin, 2004.
32. B.F. van Dongen and W.M.P. van der Aalst. Multi-Phase Process Mining: Building Instance Graphs. In P. Atzeni, W. Chu, H. Lu, S. Zhou, and T.W. Ling, editors, *International Conference on Conceptual Modeling (ER 2004)*, volume 3288 of *Lecture Notes in Computer Science*, pages 362–376. Springer-Verlag, Berlin, 2004.

33. W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.
34. W.M.P. van der Aalst, H.T. de Beer, and B.F. van Dongen. Process Mining and Verification of Properties: An Approach based on Temporal Logic. BETA Working Paper Series, WP 136, Eindhoven University of Technology, Eindhoven, 2005.
35. Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, New York, 1991.
36. A. Pnueli. The Temporal Logic of Programs. In *Proceedings of the 18th IEEE Annual Symposium on the Foundations of Computer Science*, pages 46–57. IEEE Computer Society Press, Providence, 1977.
37. A.K.A. de Medeiros, A.J.M.M. Weijters, and W.M.P. van der Aalst. Using Genetic Algorithms to Mine Process Models: Representation, Operators and Results. BETA Working Paper Series, WP 124, Eindhoven University of Technology, Eindhoven, 2004.
38. W.M.P. van der Aalst and A.J.M.M. Weijters, editors. *Process Mining*, Special Issue of Computers in Industry, Volume 53, Number 3. Elsevier Science Publishers, Amsterdam, 2004.
39. T. Winograd and F. Flores. *Understanding Computers and Cognition: A New Foundation for Design*. Ablex, Norwood, 1986.
40. T.W. Malone. Commentary on Suchman article and Winograd response. *Computer Supported Cooperative Work*, 3(1):37–38, 1995.
41. L. Suchman. Do Categories Have Politics? The Language /Action Perspective Reconsidered. *Computer Supported Cooperative Work*, 2(3):177–190, 1994.
42. T. Winograd. Categories, Disciplines, and Social Coordination. *Computer Supported Cooperative Work*, 2(3):191–197, 1994.
43. J. Begole, J. Tang, R. Smith, and N. Yankelovich. Work Rhythms: Analyzing Visualizations of Awareness Histories of Distributed Groups. In C. Neuwirth and T. Rodden, editors, *Proceedings of the 2002 ACM conference on Computer Supported Cooperative Work*, pages 334–343, New York, NY, USA, 2002. ACM Press.