Attentive Tasks: Process-Driven Information Extraction for Multichannel Documents

Kristin Stamm*, Andreas Dengel†
German Research Center for Artificial Intelligence
Kaiserslautern, Germany
Email: *kristin.stamm@dfki.de, †andreas.dengel@dfki.de

***Abstract*—The increasing amount of email data has led many companies to new challenges with their employees now having to deal with information overload while managing multiple communication channels, e.g., email, mail, and phone. Most existing approaches for reducing email processing time require significant domain specific customization efforts to achieve good performance and lack handling of attachments. We aim at providing a more domain independent approach by integrating the process context and using the information expectations of a process to guide the information extraction schedule. We rely on the concepts of *Attentive Tasks (ATs)* and *Specialist Board (SB)* from the field of document analysis. *ATs* are templates that describe all relevant and expected information about a process currently waiting for input. The *SB* provides a machine readable description of information extraction methods, so-called *specialists*, that extract all relevant information for further processes. We present our approach and demonstrate the benefits for a domain specific application, i.e., a financial institution.**

***Index Terms*—process context; email; information extraction**

I. Introduction

Since the introduction of email communication in 1975, enterprises have been registering tremendous challenges caused by this additional communication channel. One of the biggest issues, in our opinion, is the work overload involved in manually processing incoming emails and managing multichannels.

According to Belotti et al., workers' overload is caused by the increasing number of incoming emails and the complexity of email related tasks [1]. Radicati [2] forecasts a doubling of emails sent in 2013 compared to 2009 and estimates for workers an average 25% of daily work time for email processing. In many enterprises, emails are still processed manually. Taking into consideration the trend of email overload, emails are increasing the costs in enterprises.

Email, as a new communication channel to the existing mail, fax, and telephone, also increases the complexity in customer care. Since, each channel is usually handled by a dedicated system which the worker has to manage separately, enterprises seek an integrated multichannel system. To overcome the challenges of time consuming media breaks and increasing email overload, it is necessary to provide users with more support in terms of (semi-)automation in email understanding and processing, and more flexibility to the communication channel. Some approaches aim at "understanding" emails by using information extraction (IE) on the content of the email but often lack efficiency or quality. For email management, we also reproach the lack of domain independence, applicability and multichannel integration.

A major problem in email management is the insufficient handling of attachments, because most approaches focus on text analysis. Fig. 1 depicts that emails can provide all document formats - meta data, text, and images - due to the use of attachments. Especially in business environments, attachments contain important information. For example, 69% of enterprises use emails to exchange electronic invoices or bank data [3]. We, therefore, belief that enterprise oriented email management should use concepts and methods from traditional document analysis.

We aim at developing an approach that gives enough flexibility to handle the problems caused by email communication in an efficient way. Since emails in enterprises often invoke or relate to a task or a process, we suggest combining the idea of task-oriented email management with traditional IE by using the process context to guide IE. Our hypotheses are that (h1) enterprise processes have information expectations towards incoming requests and (h2) extraction results and runtime costs can be improved by integrating the process context of emails. We suggest to apply two concepts: (1) the usage of the emails' process context by introducing *Attentive Task (AT)* templates in order to better define extraction goals and to guide IE. (2) A *Specialist Board (SB)* based on the work of Dengel and Hinkelmann [4] that allows to automatically generate IE programs employing the description and evaluation of specialist methods.

In the following section, we shortly present related work. We then describe the building blocks of our approach focusing on the concepts of *ATs* and *SB*. In a next step, we explain the

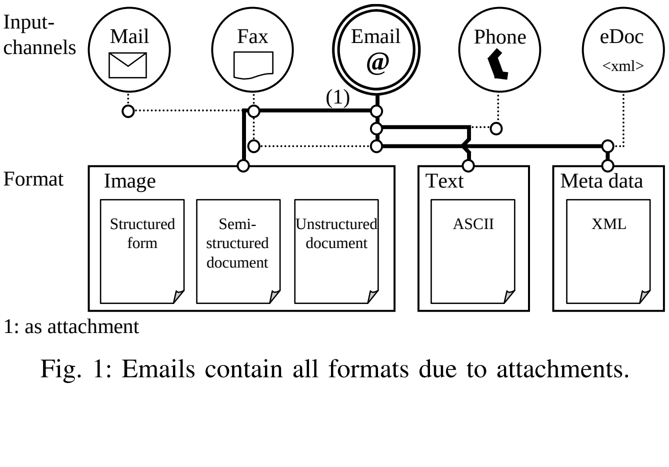

Fig. 1: Emails contain all formats due to attachments.

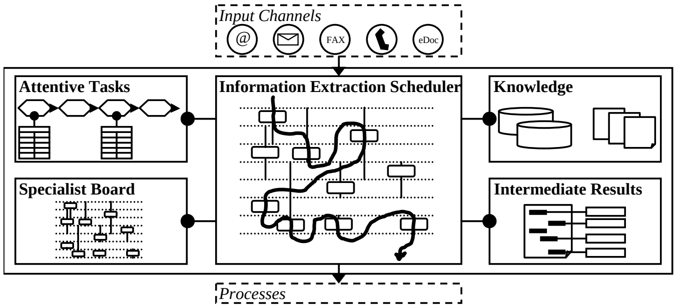

Fig. 2: Building blocks of our process-driven information extraction approach.

results of a preliminary study in an enterprise revealing the need for email management support. Based on this study, we implemented a prototype and conducted some first evaluations on extraction results and execution time. Finally, we summarize our results and conclude with tasks for future work.

II. Related work

The challenge of email management has been approached in different ways, for example, with task-centric email management or with IE management.

Task centric email management builds on the fact that most incoming emails within an enterprise environment trigger a task. It is therefore necessary to find the underlying task for each incoming email. Approaches in this research direction relate emails to tasks and use the task context to support the worker [1], [5]–[7]. Unfortunately, these approaches have a lot of drawbacks. First, they are often very domain specific and require significant customization efforts to support new domains. They further do not rely on task context to extract relevant information, are often focused on emails, and do not discuss the integration of the other communication channels.

Another research direction is the generation of IE programs. In the last decades, numerous specific IE methods have been created. These methods need to be organized sequentially in order to reach the extraction goal. These programs are often either manually designed or taught through machine learning techniques [8]. Researchers in this field are still challenged by high costs and complexity for program design and long execution times. The use of general IE frameworks like the General Architecture for Text Engineering (GATE) does not perform sufficiently in all domains [9].

Baumgartner et al. [10] and Krishnamurthy et al. [11] address these challenges by extending declarative database languages to the IE domain. This leads to shorter design phases and optimized program execution time by applying optimization techniques from database research. But the programs still need to be designed by hand for a specific domain. Supporting additional domains remains too expensive and complex for manual program design.

III. Process-driven information extraction

To solve the challenges of enterprise email management, we extend the IE approach of the *Specialist Board (SB)* introduced by Dengel and Hinkelmann [4]. The main goal of the original *SB* is to enable the automatic generation of an optimized IE plan by describing all available IE methods – the specialists – in a formalized and machine readable way. We extend the *SB* approach as follows: (1) We include information expectations from the process instances towards incoming documents. We formulate these expectations as *ATs* and use them to create a more precise extraction goal. (2) We use continuous planning to allow adaptation of the extraction plan, based on intermediate results and relevant *ATs*. (3) We apply this concept to the domain of email and multichannel management. The building blocks of our process-driven IE approach are depicted in Fig. 2. The system processes incoming documents from all input communication channels, maps them to *ATs*, extracts process-relevant information, and provides the extraction result in a structured format to the process. The IE scheduler represents the core element of our system and generates an extraction plan to reach the extraction goal. During iterative planning and extraction execution, the scheduler interacts with the remaining independent blocks: the process context in form of *ATs* that is created independently in the processes, the *SB* containing formalized information about available extraction methods, and enterprise knowledge for extraction and planning decisions, as well as a storage for intermediate extraction results for each document. In the following subsections, we discuss the role and functionalities of each block in detail.

TABLE I: Example showing the information for the attentive task LoanRequest while waiting for a process.

Descriptor	Value	Type	Constraints	Field
SenderName	Ina Mueller	Person	isCustomer	Ident.
SenderEmail	ina@mueller.org	EmailAddress	Related(senderName)	Ident.
LoanType	Dispo	LoanType	{dispo, longTerm}	Other
LoanAmount	?	Money	LargerThan(0)	New
StartDate	?	Date	After(today)	New
EndDate	?	Date	After(startDate)	New

?: New value expected; Ident.: Indentifying

A. Attentive Tasks

The purpose of our *Attentive Task (AT)* approach is to formalize information expectations towards an incoming document in the process. In enterprises, processes represent a sequence of activities necessary to achieve a certain goal. A process is often instantiated by new customer requests or interrupted due to customer interaction, especially in transactional units with high customer interaction. For example, when a customer writes an email to apply for a new loan from his bank, a ”new loan” process is invoked. The service employee might ask the customer to provide some additional information about his request, leading to an interruption of the process execution until the missing information can be found in the response of the customer. In both cases, the employee has expectations which information should be contained in the request of the customer - first general information about the loan and then more specific additional information. Based on this, we can create a template that describes the expectations

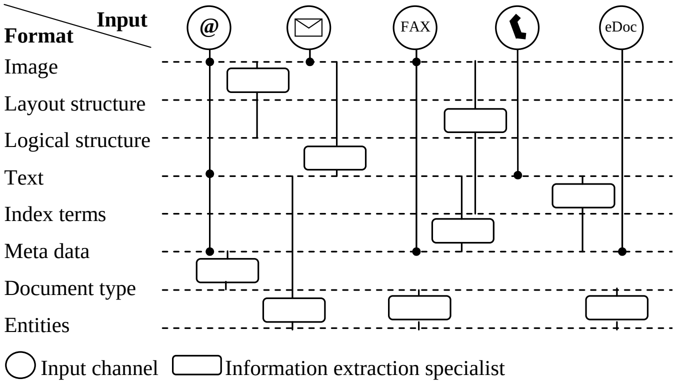

Fig. 3: The Specialist Board.

in a machine readable way. In order to release the worker from actively waiting, these templates are collected and remain active, i.e., "attentive" while waiting for the right incoming request.

We define an *AT* as a schema formalizing knowledge and information expectations by the means of slots. Some examples for such slots are given in Table I. Each slot is build by a descriptor that implies the relation to the process or the incoming document, the value of a defined information type, and a set of known constraints about the expected value. We differentiate three different kinds of fields:

- *New information field* is empty and used to describe expectations towards new information in the document later needed in the process.
- *Identifying field* contains data to help identifying the process instance, such as , "SenderName=Ina Mueller".
- *Other context field* contains a value that is not likely to appear in the incoming document.

ATs are generated independently from the IE process and all active instances are collectively available. The scheduler can search them given the currently available extraction results.

B. Specialist Board

The *Specialist Board (SB)* aims at making IE specialists methods available in a machine readable way. During the IE process, the document is transformed in different formats, e.g, from image to text or from text to layout structure as illustrated in Fig. 3. The approach categorizes specialists according to the transformation they perform. We consider the following structures: image, layout structure, logical structure, text, index terms, meta data, document type (class), and entities. More details are given in [4]

Additionally to the original approach, we consider the fact that some specialists are better suitable to the different input channels due to the information structure that they provide and other specifics. For example, mail is provided as image whereas emails additionally provide text and meta data, thus requiring different extraction methods. Channels can also vary between similar structures, e.g., fax documents have much lower quality than scanned mail documents. These criteria will therefore be included in the specialist description.

TABLE II: Example of the formal method description for the customer database match specialist.

Accessibility	Name	*CustomerDatabaseMatch*
	Type	*Specialist*
	Path	*Information.extraction.DBMatchMethod*
Planning Information	Precondition 1	*isText(span)*
	Precondition 2	*DatabaseAvailable(db_customer)*
	Postcondition	*CustomerIdentified(span)*
Parameters	Input 1	*FieldList:customerFields*
	Input 2	*String:span*
	Input 3	*List:weightOfCols*
	Output	*FieldList:customerField*
Suitability	Quality	*f(precision, recall)*
	Costs	*f(runtime, storageSpace, nbAnnotations)*

Table II contains an example of a formal description framework for a specialist. It describes the *accessibility* to the specialist, information to enable automatic *planning*, input and return *parameters* to use the method correctly, as well as information about the *suitability* under given circumstances. To access a specialist, we need name, type, and path to the method. For automatic planning purposes, we need to define which preconditions and postconditions need to be met before and after executing the method. This includes especially information about available information structures and properties depending on the communication channel. Further, we need to define which input parameters need to be transferred to the method and which variables are expected to be returned. The suitability measure function is crucial for the decision between several similar methods and can include different aspects depending on the goals of the system. We consider assumptions about the extraction quality, such as precision and recall, towards the defined extraction goal and efficiency measures, like expected runtime or used storage.

C. Knowledge

Similar to the worker realizing email processing manually, we need to access additional enterprise knowledge about customers, business partners, and contracts during the IE execution. We use this knowledge to verify extraction results and to decide about the further extraction steps. This knowledge can be made available in databases, documents or via interfaces to other enterprise systems.

D. Intermediate results

Intermediate extraction results for each document need to be stored and made available to the scheduling module and to specialists that are currently executed. One possible format is the document structure used by the GATE system [9]. A document is stored in its original format and each extraction result is stored as annotation to a text span. Additionally, we need to track all information about the current state and the goal state. The extraction result is similar to the *AT* structure and consists of a set of fields. For each of these fields we maintain a confidence index about the extracted value.

E. IE Scheduler

The scheduler is the core element of our system and uses all available context information in order to plan, execute, and replan IE until all information required in the following

TABLE III: Exemplary analysis of 48 documented processes.

Expected Information	Invoke	Inv./Wait	Wait	Other	Sum (%)
Identifying (exist)	–	26	2	–	28 (58%)
Identifying (new)	14	26	–	–	40 (83%)
New	6	18	1	–	25 (52%)
Overall	14	26	2	6	48
(%)	(29%)	(54%)	(4%)	(13%)	(100%)

process has been extracted. The algorithm breaks down into the following steps:

1) Get the document. Initialize current state and initial empty extraction plan.
2) Prioritize available *ATs* with *Dempster Shafer's Rule* according to the existing evidences.
3) If all *AT* priorities are below threshold, select new *AT* template and instantiate.
4) Define extraction goal based on current knowledge, i.e., highest prioritized *ATs*, other available information about input channel.
5) Generate extraction plan by using *Continuous Partial Order Planning* and the *SB*.
6) Execute next extraction method.
7) If extraction goal reached, return results, else goto (2).

IV. Application in enterprises

Since our main goal is applicability in enterprises, we discuss in this section the relevancy of our approach in enterprises and evaluate the approach in an enterprise motivated test environment. We therefore conducted a preliminary study and evaluations: A process review in a financial institution to examine information expectations in processes. Following, we generated a corpus with test persons to conduct first evaluations with a prototype.

A. Information expectations in business processes

This preliminary study helps understanding the relevancy of our *AT* approach and was carried on a German financial institution with over 5,000 employees world wide. We focused on a service center where employees mainly interact with customers. We analyzed 48 of the organization's processes that were already documented in text format and available to the employees via intranet. For each process, we identified if it was invoked by an external request (Invoke), if there were activities waiting for external response (Wait), or both (Invoke/Wait). We further examined the information types expected in the incoming request similar to the *AT* fields, i.e., new or existing *identifying* information and *new* information for the enterprise. The main findings of the process analysis are summarized in Table III:

- *Input channels as trigger.* Input channels are the main trigger of processes in this unit. 83% (29%+54%) of the processes are invoked by a request from an external input channel and 58% (54%+4%) have at least one activity that is waiting for a response through an input channel.
- *Information expectations in processes.* Most processes (83%) expect new identifying information at the process instantiation and still 58% use this information later to identify the process instance. 52% of the processes expects new, unknown information from incoming requests.
- *Relevancy of multichannel management.* An additional analysis of requests per communication channel shows that currently telephone (54%) and mail (37%) are the main input channels whereas email is used with only 9% and fax with 1%. In interviews, employees predicted an increase in the email channel and complained about the different systems they have to use for each channel.

We conclude that the process-driven IE approach would be applicable and helpful for this kind of organization, allowing the integration of external communication into the internal processes. We have seen information expectations in the process descriptions towards incoming documents. In the next sections, we explain how the application of our approach could be realized and evaluate a first implementation of our concepts.

B. Corpus

A cohesive corpus including email communication threads between customers and a company, and *ATs* expressing the expected information in the related process are required to evaluate our approach. Since no corpus is currently available, we generated in a first step a test corpus. We selected two processes from out financial institution and asked participants to play the role of ten customers and two service employees. All participants have personal experience with financial institutions and email communication.

The customers had to perform two tasks: (1) Change the owner of their contract. (2) Postpone payments to a new deadline. Based on a brief task description and some fictive contract information, customers had to send email requests to the service center. The service employees reacted to incoming emails according to two process descriptions using answer templates – both based on our case study partner's processes. During two weeks, customers created a corpus of 48 emails: 18 process invokes and 29 with additional information for existing processes. Due to the open task description, first emails lack in most cases information to proceed. The customers, therefore, needed to provide additional information in the next email. We generated *ATs* based on provided and missing information during the conversations.

C. Prototype

We implemented a first prototype in Java to evaluate our process-driven approach. We defined *ATs* and *SB* descriptions in XML format. The IE specialists are a set of standard A Nearly New Information Extraction System (ANNIE) methods from GATE [9] and self-implemented extraction methods.

Active *ATs* are generated and stored manually in a central folder. The prototype can process emails and their attachments stored in a central inbox folder. The IE schedule can be predefined as a fix pipeline or be generated dynamically during runtime according to the corresponding *ATs*.

D. Experimental setup

The goal of our first evaluations is to better understand how process-driven IE influences performance. We compare our approach with a general IE framework and brute force extraction, i.e., execution of all available methods.

The experiments have been conducted on the corpus described previously with four IE scheduling methods:

- *GATE (fix):* Fixed execution of the standard ANNIE pipeline including Default Tokeniser, Default Gazetteer, Sentence Splitter, Part of Speech Tagger, Transducer, OrthoMatcher, and Coreferencer. This pipeline represents a domain independent framework and should reveal how well these frameworks do perform in new domains.
- *GATE + specialists (fix):* Since we expect extracted information of the GATE pipeline to be insufficient we extend the GATE methods with specialist methods, e.g., regular expression extractor, database matcher, and classifier. This fix order pipeline represents a full tool set to extract all relevant information.
- *Specialists (fix):* Execution of all domain specialists in fixed order.
- *Specialists (dynamic):* Dynamic IE with specialists based on fields in the corresponding *AT* to evaluate quality and cost performance of the process-driven approach.

We determine for the evaluations precision Pr, recall Re, and f1 measure $F1$, as well as runtime cost C per data size for an IE pipeline p on a document d as follows:

$$Pr_{p,d} = \frac{|A_d^{rel} \cap A_{p,d}^{ex}|}{|A_{p,d}^{ex}|}, Re_{p,d} = \frac{|A_d^{rel} \cap A_{p,d}^{ex}|}{|A_d^{rel}|}$$

$$F1_{p,d} = 2\frac{Pr_{p,d}Re_{p,d}}{Pr_{p,d} + Re_{p,d}}, C(p,d) = \frac{run_{p,d}}{s_d}$$

Where $A_{rel}(d)$ are all process-relevant (ATs) annotations in the document, $A_{ex}(p,d)$ all extracted annotations (excluding intermediate results), run the runtime, and s the file's size.

E. Evaluation results

The evaluation of the different IE schedules on our test corpus has revealed strong differences in performance. Table IV contains the average results for each scheduling method.

IE with the fix GATE pipeline shows very low precision of 13% and low recall of 48%. The precision value is caused by a large amount of extraction results not relevant for the corresponding process. The recall value indicates that the available methods in GATE are not sufficient to extract all information for our special domain. We therefore had to implement additional domain specific methods that meet the processes' expectations. The GATE pipeline shows very high runtime costs (594 μs/byte) due to many unnecessary extraction steps. Extending the GATE pipeline with domain specific specialists leads to better recall (98%). Since the problem of irrelevant extraction results has not been changed, precision and runtime are not much improved. Executing all specialists methods in a pipeline reduces irrelevant extraction results tremendously and leads to better precision (88%) and runtime results (12 μ/byte). The dynamic scheduling of IE methods shows additional improvements in precision (90%).

TABLE IV: Performance evaluations IE schedule methods.

IE scheduling method	Pr	Re	$F1$	C^1
GATE (fix)	13%	48%	0.19	594
GATE + specialists (fix)	17%	98%	0.27	598
Specialists (fix)	88%	98%	0.92	12
Specialists (dynamic)	90%	98%	0.93	14

1: in μs per byte

First evaluation results show that process-driven IE can improve both quality and costs:

1) *Introduction of specialists* help achieving best performance since standard IE frameworks do not suffice.
2) *Reduction of unnecessary extraction results* since *ATs* realize specific extraction for both the fix and dynamic specialist schedule approaches.
3) *Optimization of extraction costs* since using *ATs* optimizes runtime.
4) *Limited use of general frameworks* to overcome lacks in domain specific specialists.

V. Conclusion and future work

We combined the concepts of *Attentive Tasks* and *Specialist Board* to address enterprises' challenges of email overload and multichannel management. Evaluations for a first case study helped us identify information expectations in customer related processes and demonstrate how process-driven dynamic IE planning can improve precision, recall, and runtime costs.

Further investigations are, however, required to validate these first results. In a next step, we plan to repeat the experiments on a larger test corpus in cooperation with our case study partner. Also, we will need to extend the specialist tool set with more methods, e.g., support vector machine classifiers and more regular expression extractors. We further plan investigations on the planning algorithms, the influence of incomplete specialist tool sets, and the improvements of our approach regarding multichannel management. On the longterm, we plan further user studies to determine which *AT* based approach improves email management best.

References

[1] V. Bellotti *et al.*, "Quality vs. quantity: Email-centric task-management and its relationship with overload," *Human-Computer Interaction*, vol. 20, pp. 1–2, 2005.

[2] Radicati, "Email statistics 2009-2013," The Radicati Group Inc., Tech. Rep., 2009. [Online]. Available: http://www.radicati.com/?p=3237

[3] P. Schmitter, "Rechtliche und technische fragen im e-mail-management," 2005. [Online]. Available: \url{http://www.documanager.de/magazin/artikel_608-print_rechtliche_und_technische_fragen.html}

[4] A. Dengel and K. Hinkelmann, "The specialist board a technology workbench for document analysis and understanding," in *IDPT, 2nd World Conference on Design & Process Technology, Austin, TX, USA*, vol. 2, December 1996, pp. 36–47.

[5] M. Dredze, T. A. Lau, and N. Kushmerick, "Automatically classifying emails into activities," in *IUI*, 2006, pp. 70–77.

[6] N. Kushmerick, T. A. Lau, M. Dredze, and R. Khoussainov, "Activity-centric email: A machine learning approach," in *AAAI*. AAAI Press, 2006.

[7] J. Gwizdka, “Reinventing the inbox: supporting the management of pending tasks in email,” in *CHI’02 extended abstracts on Human factors in computing systems*. ACM, 2002, pp. 550–551.

[8] S. Sarawagi, “Information extraction,” *Foundations and Trends in Databases*, vol. 1, no. 3, pp. 261–377, 2008.

[9] H. Cunningham *et al.*, *Text Processing with GATE*, version 6 ed., University of Sheffield Department of Computer Science, April 2011. [Online]. Available: http://gate.ac.uk/

[10] R. Baumgartner, S. Flesca, and G. Gottlob, “Declarative information extraction, web crawling, and recursive wrapping with lixto,” *Logic Programming and Nonmotonic Reasoning*, pp. 21–41, 2001.

[11] R. Krishnamurthy, Y. Li, S. Raghavan, F. Reiss, S. Vaithyanathan, and H. Zhu, “Systemt: a system for declarative information extraction,” *ACM SIGMOD Record*, vol. 37, no. 4, pp. 7–13, 2009.