Representation and presentation of knowledge and processes – an integrated approach for a dynamic communication-intensive environment

Received 25 March 2021
Revised 8 July 2021
Accepted 10 July 2021

Gerd Hübscher
Hübscher and Partner Patentanwälte GmbH, Linz, Austria and Polymind GmbH, Vienna, Austria

Verena Geist
Software Competence Center Hagenberg GmbH, Hagenberg, Austria

Dagmar Auer
Institute for Application-oriented Knowledge Processing and LIT Secure and Correct Systems Lab, Johannes Kepler University Linz, Linz, Austria

Nicole Hübscher
University of Art and Industrial Design Linz, Linz, Austria, and

Josef Küng
Institute for Application-oriented Knowledge Processing and LIT Secure and Correct Systems Lab, Johannes Kepler University Linz, Linz, Austria

Abstract

Purpose – Knowledge- and communication-intensive domains still long for a better support of creativity that considers legal requirements, compliance rules and administrative tasks as well, because current systems focus either on knowledge representation or business process management. The purpose of this paper is to discuss our model of integrated knowledge and business process representation and its presentation to users.

Design/methodology/approach – The authors follow a design science approach in the environment of patent prosecution, which is characterized by a highly standardized, legally prescribed process and individual knowledge study. Thus, the research is based on knowledge study, BPM, graph-based knowledge representation and user interface design. The authors iteratively designed and built a model and a prototype. To evaluate the approach, the authors used analytical proof of concept, real-world test scenarios and case studies in real-world settings, where the authors conducted observations and open interviews.

The research reported in this paper has been partly funded by BMK, BMDW, and the State of Upper Austria in the frame of the COMET Program managed by FFG, by the LIT Secure and Correct Systems Lab funded by the State of Upper Austria and by the FFG BRIDGE project KnoP-2D (grant no. 871299).

International Journal of Web Information Systems
Vol. 17 No. 6, 2021
pp. 669-697
Emerald Publishing Limited
1744-0084
DOI 10.1108/IJWIS-03-2021-0031

Findings – The authors designed a model and implemented a prototype for evolving and storing static and dynamic aspects of knowledge. The proposed solution leverages the flexibility of a graph-based model to enable open and not only continuously developing user-centered processes but also pre-defined ones. The authors further propose a user interface concept which supports users to benefit from the richness of the model but provides sufficient guidance.

Originality/value – The balanced integration of the data and task perspectives distinguishes the model significantly from other approaches such as BPM or knowledge graphs. The authors further provide a sophisticated user interface design, which allows the users to effectively and efficiently use the graph-based knowledge representation in their daily study.

Keywords Knowledge work, Business process management, Human communication, User interface concepts, Graph-based knowledge representation

Paper type Research paper

1. Introduction

Companies and other organizations today are faced with highly dynamic, competitive environments. The digitization of communication-intensive knowledge work is not only one of the greatest challenges but also one of the greatest opportunities. Knowledge workers such as researchers, developers, consultants or lawyers heavily rely on communication. From a knowledge perspective, observable communication data is mostly unstructured and requires a-priori knowledge to extract semantic concepts.

The lack of a model that allows to define and handle *mental concepts* [1], which evolve during daily work, hinders a continuous process of knowledge-applying data transformation tasks of individual users. This might be one reason that business process modeling, as well as individual and organizational learning processes, have not yet been successfully applied to data- and knowledge-driven, process-oriented knowledge work.

The need for a *high degree of flexibility* for non-routine, problem-solving tasks does not fit with traditional business process management (BPM) systems, which mainly adopt a top-down approach for predefined administrative processes while knowledge management (KM) systems lack the task context. We regard processes as a sequence of tasks. Furthermore, in an organizational setting not only knowledge work but also *standardized processes* within the same context need to be supported, e.g. legal constraints or compliance rules that clearly define procedures and also their tracing. Thus, both – highly dynamic processes with a strong focus on communication and predefined, well-structured ones need to be supported within one overall system.

We consider a flexible and adaptable, bottom-up approach to integrated knowledge and process management, relying on an appropriate model as a promising approach. Therefore, our research focuses on a model of integrated data and task perspectives, a seamless bottom-up approach for the continuous maturing of these perspectives and the suitability of this approach in real-world settings with a special focus on user-experience.

In this paper, we propose an easily adaptable knowledge- and process-centered model for the exchange of data between knowledge workers in flexible and not only adaptable business processes but also in well-defined administrative processes. As the building blocks of the model are strongly linked, a graph-based representation of knowledge and process artifacts is considered. Non-observable information artifacts are derived from observable, communicated data along with different dimensions as follows:

- Knowledge in the form of structured and unstructured data, documents, graphics and information extracted and stored in a graph; and

- The dynamic behavior, i.e. data-centric processes that evolve in the graph when knowledge is created.

To support efficient and effective use of the proposed TE𝔸M model, we provide a sophisticated user interface design which considers the high complexity of the graph-based knowledge representation.

The further paper is structured as follows. In Section 2 we briefly present our overall research methodology and provide the real-world example which we use throughout this paper. Section 3 provides-related work in the area of knowledge work, flexible and adaptable support for business processes, knowledge graph and user interaction modeling. Our proposed model and modeling approach is presented in Section 4. In Section 5, we discuss our TE𝔸M model from a knowledge graph viewpoint. Section 6 provides insights into the user interface concept, which supports users to handle the high complexity of the underlying graph-based model. Our running example is further visualized with screenshots. We discuss our most important results in Section 7. The paper concludes with a summary and an outlook on future work in Section 8.

2. Research methodology

We follow a *design science research methodology* (Hevner *et al.*, 2004; vom Brocke *et al.*, 2020), which relies on iteratively evaluating intermediate results. The environment for our research is patent prosecution in the patent law firm of one of our project team members. It is characterized by a highly standardized, legally prescribed process on the one hand and individual knowledge work when translating new technical knowledge into legally binding language on the other hand. Furthermore, this work requires intensive communication, within an organization, as well as with clients and external partners. However, the current information technology (IT) infrastructure consists of rather isolated tools, without integration of processes and data.

The following *real-world example* is used throughout this paper, to support the discussion of our approach.

It is also a representative example for the test scenarios and use cases studied to evaluate our results. The starting point of this example is the receipt of an e-mail from a client with two distinct concerns as follows:

(1) It contains the notification that the address of one patent proprietor has changed; and

(2) The client requests whether, in view of the first office action received, it is appropriate to pursue a certain patent application.

The reply is urgently expected because the time limit for filing observations in reply to the office action expires in a few weeks.

Thus, this e-mail initiates two different actions as follows: It cannot be assumed that the e-mail is already an order to enter the address change in the relevant official registers. Certain facts need to be clarified before as follows: property rights with the indicated patent proprietor as the owner, costs of the changes and finally a professional assessment of whether such a change should be indicated before the relevant offices at all. Furthermore, this change potentially not only affects a single case but also several ones. Requires to study the related case file, as well as the office action, to procure the prior art cited therein, to exam the citation and to elaborate one or more means of defense. Experience and knowledge of the practice of the patent office are needed for this. Furthermore, it is questionable whether the stated urgency applies at all, whether extensions of time limits are possible or whether alternative prosecution routes are available. Finally, an evaluation of the existing possibilities and a concrete recommendation for action is required. As all actions can have serious legal consequences, in particular tracing of actions and decisions based on the collected information is needed.

Throughout this paper we will focus on different aspects of this scenario such as structuring data, reusing data, relating data to several tasks, tracing exchanges and flexibility in the overall model evolution.

The contributions of our work to solve the problems identified before are as follows:

- A highly flexible and adaptable, user-centered model for integrated knowledge and process management and its prototypical implementation using a graph database;
- A layered modeling approach that allows for flexibility while providing a certain level of guidance to support individual users and traceability of communication; task handling and data manipulation;
- A user interface concept which supports working on the complex integrated model graph; and
- A demonstration case of applying the model in a real-world setting, i.e. in the domain of patent prosecution, which already shows promising results.

The results are described in detail in Sections 4–7. To evaluate the concepts, models and prototypes built during this project, we focused on an analytical proof of concept in the initial phases, involving external experts into feedback circles and discussions. Based on these results, the first prototypes were implemented to provide proof by construction with selected real-world scenarios. To involve potential users, case studies have been conducted in real-life environments, not only testing the core model but also especially user interaction. The users were partly observed and also encouraged to individually play around with the prototype. Besides their produced results, especially the open interviews with the users provided valuable insights. The user interface design has been further supported by a pre-study, where requirements have been identified and alternative user interface concepts have been tested and discussed in a target group with user interface sketching.

3. Related work

Different areas of research, such as knowledge work, business processes and their automation, document management and user interaction modeling, have a strong impact on our work.

Drucker (1959) characterizes knowledge work by its adaptable and creative nature. It is strongly related to KM (Dalkir, 2005), which is a multi-disciplined approach promising a high benefit of knowledge for an organization (McElroy, 2003). In practice, KM (Bailey *et al.*, 2010; Drucker, 1999) is often criticized to provide few benefits but requiring high effort. Even though large knowledge bases are available now, it is not straightforward to find the right information and often results in intensive search processes.

While traditional knowledge work is heavily relying on the information, increasing complexity in most work domains requires enhanced support for human communication. As KM is about processes dealing with knowledge (e.g. acquiring, creating and sharing knowledge), the three main interpretations include as follows: combining KM activities with an independent process (Gronau and Weber, 2004; Papavassiliou *et al.*, 2002), the management of knowledge-intensive business processes (Karagiannis and Telesko, 2000; Marjanovic, 2005) and the integration of knowledge-intensive activities into operational business processes. These approaches are all rooted in the workflow tradition, i.e. an activity-centric approach. None of them had much impact, neither in research nor in practice.

BPM (Dumas *et al.*, 2013) is typically associated with well-structured, highly predictable and, thus pre-definable business processes. Such processes show a high number of repetitive activities and have been subject to automation for a long time (Weske, 2012; Ter Hofstede *et al.*, 2010). Common activity-centric BPM is often missing support for user interaction modeling in the context of business processes and does not offer an appropriate data model (Auer *et al.*, 2009). Trætteberg (2008) identifies domain and data modeling as one of the weakest points of business process modeling languages, such as business process model and notation (BPMN). The increasing need to provide adequate support for knowledge work demands new approaches in BPM. The focus is no longer on "what must be done," but rather on "what can be done" (Van der Aalst *et al.*, 2005), enabling knowledge workers to prepare working plans at run-time based on their knowledge and needs and to take decisions within the working context.

Different strategies are used to deal with these shortcomings. However, despite the active area of research on flexible and adaptable business processes, business process modeling has not yet been successfully applied to data- and knowledge-driven, process-oriented knowledge work.

One direction of research is to extend activity-centric approaches with special features to allow for more flexibility (Reichert and Weber, 2012; Döhring *et al.*, 2011; Hallerbach *et al.*, 2010), which enable some run-time planning – but only within a predefined frame. Rule-based and constraint-based declarative models offer a higher level of flexibility than traditional activity-centric approaches (Pesic and van der Aalst, 2006; Reichert and Weber, 2012). However, many of these approaches are generating process models from predefined specifications, e.g. in the SDeclare language (Jiménez-Ramírez *et al.*, 2015), still focusing on control flow and rarely taking the data perspective into account. Another approach in the field of variability modeling is based on graph transformation techniques (Geist *et al.*, 2016) to deal with the exponentially increasing number of business process variants; however, again the definition of a base process model is required.

Flexibility and adaptability are often associated with data-driven approaches to business process modeling and execution. The key driver for these processes is no longer a predefined control flow, but the availability of data. Representatives of this approach are, for example, case management (Marin *et al.*, 2013; OMG, 2014; Van der Aalst *et al.*, 2005) or object-aware processes (Künzle, 2013; Künzle and Reichert, 2011). The most important drawback of case management is the focus on predefined process types, determining the allowed planning frame at run-time, which is often a limiting factor for knowledge workers' variously changing, specific and creative work. Object-aware processes, on the other hand, are not designed for adaptability and dynamic model evolution at run-time.

In knowledge work, information is often coarse-grained (e.g. documents) and needs to be processed to extract fine-grained information (e.g. structured and semi-structured, related data). Common information and document management systems, by contrast, support some kind of tagging, but they are not able to preserve the context when extracting information, e.g. from an e-mail. This demand for integrating an underlying semantic model, where processable knowledge can be modeled and stored using a network-oriented representation in the form of human tasks consuming and producing data artifacts. Such semantic-based modeling techniques (Domingue *et al.*, 2011; Davies *et al.*, 2009) to business process specification allow to independently reuse the extracted data with respect to their original context, which establishes a task context (similar to the notion of a shopping cart) for executing unpredictable, collaborative processes.

Earlier works in the area of semantic BPM (Hepp *et al.*, 2005; Hepp and Roman, 2007) focus on combining semantic web services with BPM technology for supporting agile process implementation. However, the perspective is a rather technical one, focusing on the lack of machine-accessible semantics but not taking into account dynamic communication between knowledge workers.

Knowledge graphs Pujara *et al.* (2013) and Paulheim (2017) can be applied to flexibly tie together domain knowledge from different source data structures and organizational knowledge about the team of involved humans with operational tasks. Of particular importance are the structured concepts of *data*, *information*, *knowledge* and *wisdom*, commonly known as the DIKW hierarchy (Frické, 2019). Within this hierarchy, each concept is related to the preceding concept, building a chain with increasing connectedness and understanding (Ahsan and Shah, 2006; Zins, 2007).

In cognitive sciences, one way of explicating how humans perceive their current work situation and make decisions based on previous knowledge is investigated by mental model theory (MMT) (Johnson-Laird, 1980). Mental models have also been applied in KM to learn from shared experiences in an organization (Firestone and McElroy, 2012).

A more recent related approach for designing digital work is suggested by Oppl and Stary (2019) by integrating subject-oriented business process management (S-BPM) and MMT. In contrast to our vision of a highly flexible and adaptable user-centric system, where knowledge workers shall be enabled to prepare working plans at run-time and to take decisions within the dynamic working context, this approach defines explicit process models (top-down approach). An opportunistic approach to BPM, called opportunistic business process modeling (oBPM) (Grünert *et al.*, 2014), supports bottom-up model creation, though, the focus is rather on document-centric and object-oriented artifact specification but not on human communication and process-oriented knowledge work.

4. The TE$\mathbb{A}$M model

Aiming at a seamless, mature and integrated knowledge and process management, we must start at the data and task level. In this context, we consider a process as a sequence of tasks while knowledge is represented as relations between the model components.

As a consequence, the overall vision for the TE$\mathbb{A}$M model (integrated data and task multi-dimensional graph) is to support people to effectively and efficiently work on their predefined administrative tasks, as well as on flexible and adaptable knowledge-intensive ones. A sequence of tasks is considered as a process. Besides the integration of data and tasks, communication needs to be considered to seamlessly mature integrated knowledge and process management.

4.1 Three-layer architecture

The model of the TE$\mathbb{A}$M model follows a graph-based meta-modeling approach. The multiple dimensions of the approach are represented by data and task components, which are related to each other by task relations. Furthermore, a fine-grained representation of data is considered by relating data components to each other via data relations. To allow for seamless knowledge and process maturing, starting with only a small set of basic types and instances, a highly flexible and adaptable model is needed, which can be easily enriched and adapted by qualified users from the application domain. Therefore, we do not rely on rather stable data types for the TE$\mathbb{A}$M model, but on a schema-free approach using types for classification. When installing the system, the domain model will only contain some core types, such as *NaturalPerson*. All others will be defined by experienced users during their daily work. As we follow a meta-modeling approach, these types are described by data and can easily be added, changed, etc. during run-time.

The TE𝔸M model has a three-layer architecture as follows: the *meta model*, the *domain model* and the *instance model*. All of these models reflect the data and task perspectives. Figure 1 gives an overview of the models and their dependencies.

4.1.1 Meta model. Describes the core characteristics of the domain model and the instance model, i.e. the building blocks and not only their attributes but also core constraints. The *meta model of the domain model* defines the four basic kinds of type-level concepts, i.e. data object type, data object type relation, task type and task type relation. Furthermore, the attributes to describe each of these components are specified, such as code (i.e. the name of the type), isAbstract, isPrimitive, isExtendable, created, lastChanged or number of Instances. Also, constraints can be defined in the meta model, for example, a task type relation only connects a task type and a data object type or data object type relation. The *meta model of the instance model* defines the four basic kinds of instance concepts, i.e. data object, data object relation, task and task relation. Furthermore, the describing attributes for each of these kinds of instances are defined, such as type, created, lastChanged, status History, is Validated or value. The attribute *type* is intended for reference to the classifying type, e.g. for a data object to its classifying data object type.

4.1.2 Domain model. Defines the domain-specific types and their relations. These types do not correspond to those in programming languages but are mental concepts used for classifying instances. The types are constantly evolving while the TE𝔸M model is used for daily work. Also, domain-specific constraints can be specified, for example, on data prerequisites, cardinalities or ordering.

4.1.3 Instance model. Holds all instances of the TE𝔸M model. Each instance is classified by a domain-specific type (domain model).

In the following, we discuss the core components data and tasks with their relations in detail, before dealing with their integration.

Knowledge workers and administrative users constantly encode and decode a series of symbols to or from mental concepts, when communicating information and knowledge. Mental concepts are not only created for single instances, e.g. the natural person John Doe but also for sets of instances, in the sense of classification, e.g. *NaturalPerson*. Therefore, we consider mental concepts on the type (i.e. classification) and instance level.

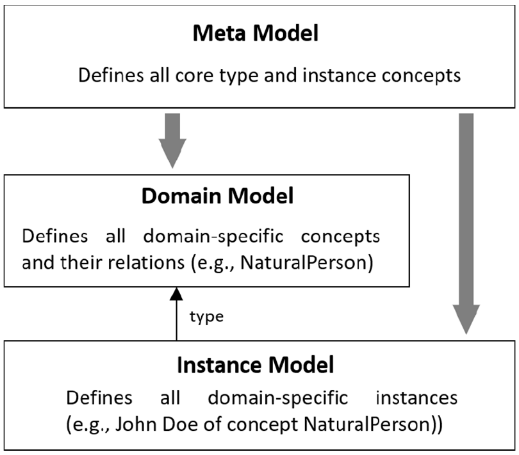

Meta Model
Defines all core type and instance concepts
Domain Model
Defines all domain-specific concepts and their relations (e.g., NaturalPerson)
type
Instance Model
Defines all domain-specific instances (e.g., John Doe of concept NaturalPerson))

Figure 1. Overview of the three layers of the TE𝔸M model

In the following discussion of the meta model components, we start with a data-oriented perspective, focusing on data components and their relations in Section 4.2. The dynamic perspective, i.e. tasks and their integration with data, is discussed subsequently in Section 4.3. In the further presentation of the TE𝔸M model we use Latin letters for instance abbreviations and Greek ones for types.

4.2 Data components and their relations

Even though much information is stored in unstructured or semi-structured documents, we strive to extract a fine-grained representation of the data. To increase data quality and to build consistent process chains, each entity exists only once in the instance model, e.g. if several people have the same first name, they all reference the same single instance.

The type-level components for data are *data object type (δ)* and *data object type relation (ρ)*, which describe the concepts used to classify the instance-level components. The instance-level components are data object (d) and data object relation (r). Both of them are continuously evolving when working with the system.

Definition 4.1. – A data object (d) represents the mental concept for an instance, e.g. the natural person John Doe. Each data object is unique within the TE𝔸M model. Two different kinds of data objects are distinguished – observable and non-observable data objects. An observable data object contains an observable value, e.g. a data object (of data object type String) immediately holds the string value. In contrast, a non-observable data object is a root of a sub-graph, which further describes the data object. The non-observable data object does not contain an observable value.

Definition 4.2. – A data object type (δ) describes a concept that classifies a set of data objects, e.g. natural person or address.

Data objects and data object types are further described by attributes. Data object types define characteristics, such as the name of the type or rules used to define different kinds of constraints shared by all data objects classified by this type. Data object attributes differ as they deal with run-time aspects, such as status history or the value of observable data objects, i.e. transactional data. Partonomical relationships are not described by attributes, but by relations. Therefore, entropy decreases along with the direction of these relations.

Definition 4.3. – A data object relation (r) can be either a partonomical (with the kinds has and *hasValue*) or an associative (with kind role) relation (thus directed) between two data objects.

Non-observable data objects are always connected to their containing, superordinate data objects via data object relations of the kind has, observable ones need the kind *hasValue*. A data object relation of kind role links a data object to its role. The direction of the relation is from role to object, i.e. the semantics of isRoleOf.

The corresponding concept on the type level is the data object type relation. The partonomical relations of the kind has and *hasValue,* the associative of kind role, as well as the generalization via the is relations are used to build the concept hierarchy.

Definition 4.4. – A data object type relation (ρ) is a directed type-level relation, with a data object type as its source and target and is further specified by its kind (taxonomical, partonomical, associative or generalization).

Within the TE𝔸M model, we have two connected semantic sub-graphs, the instance graph (i.e. instance model) and the type graph (i.e. domain model), integrated via type relations. To reduce the complexity of the further presentation of the data perspective of our model, we focus on the instance model.

An observable data object d_i^* has exactly one unique value that can be observed, e.g. in messages. Its value can be of any primitive data type. For example, the value of d_0^*, which is an observable data object representing data of type String, is “John” in Figure 2. Non-

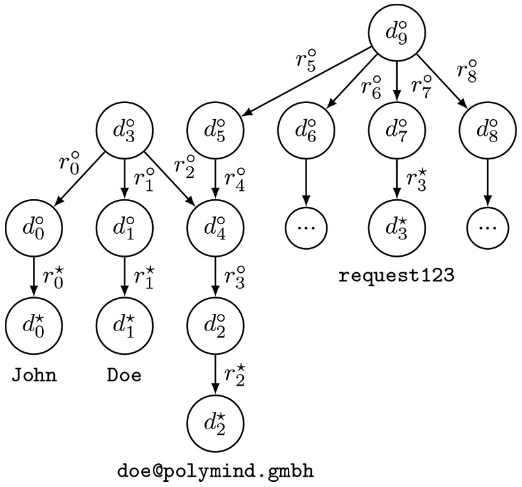


Figure 2.
$G_D = (D \cup R)$ with observable $d_i^\star$ and non-observable d_i° data objects

observable data objects d_i° have no value. They stand for abstract mental concepts which are not only represented (encoded) by related observable but also non-observable data objects (e.g. d_3° in Figure 2). Both, observable d_i^* and non-observable d_i° data objects are discrete, unorganized and have no specific meaning. The two sets of data objects are defined in as follows equation (1):

$$D^* = \{\mathrm{d}_0^*, \ldots, \mathrm{d}_n^*\} \quad D^\circ = \{\mathrm{d}_0^\circ, \ldots, \mathrm{d}_n^\circ\} \quad D = D^* \cup D^\circ \tag{1}$$

$$R^* = \{r_0^*, \ldots, r_n^*\} \quad R^\circ = \{r_0^\circ, \ldots, r_n^\circ\} \quad R = R^* \cup R^\circ \tag{2}$$

Meaning, i.e. knowledge (Section 5), is added by the relations. We can partition the edges R into two sets R^* and R° [equation (2)]. R^* connects observable data objects D^* and non-observable ones D° while R° only connects non-observable data objects D°. Those disjoint independent sets contain the vertices D (i.e. data objects) of a directed graph $G_D = (D, R)$, interconnected by the edges R (i.e. data object relations) [equation (3)] as follows:

$$G_D = D \cup R \tag{3}$$

The instance graph G_D in Figure 2 describes selected details of the non-observable data object d_9° (e-mail of John Doe with all its components). Starting with the observable data objects D^*, a bottom-up approach is used to build enriched data objects along with a path of aggregating *hasValue* and *has* relations, as well as the associating role relations (e.g. r_2°).

Thus, each element is finally assigned to an observable data object $\mathrm{d}_0^* - \mathrm{d}_3^*$, which relates to one or more non-observable data objects $\mathrm{d}_0^\circ - \mathrm{d}_9^\circ$. The values of the observable data objects are also added in Figure 2. The details for the data objects in Figure 2. i.e. its type and value, are stated in Table 1.

Table 1.
Details for the data objects *d* in Figure 2

Id	Data object type	Value
$d_0^{\star}$	String	"John"
$d_1^{\star}$	String	"Doe"
$d_2^{\star}$	String	"doe@polymind.gmbh"
$d_3^{\star}$	String	"request123"
d_0	FirstName	–
d_1	LastName	–
d_2	EMailAddress	–
d_3	NaturalPerson	–
d_4	EMailContact	–
d_5	Sender	–
d_6	Receiver	–
d_7	Subject	–
d_8	DateReceived	–
d_9	EMail	–

4.3 Task components and their relations

Besides the data components and their relations, the tasks and the relations between data objects/data object relations and tasks build the overall integrated, dynamic model. Thus, it is no longer just data enriched with semantics, but especially information and knowledge within a dynamic context.

Definition 4.5. – A task (t) represents an instance of an atomic activity of information processing.

Definition 4.6. – A task type (ξ) describes a concept that classifies a set of tasks, e.g. extracting sender information from an e-mail.

Similar to the data perspective, all entities of the task perspective have attributed at type and instance levels.

Definition 4.7. – A task relation (y) is a bipartite, directed relation between a task and a data object or data object relation.

Definition 4.8. – A task type relation (υ) is a bipartite, directed type-level relation between a source and a target type with a specific kind ^{k} (e.g. specifying data dependency, the validity of data objects or relations, user action on the task, etc.). Source and target types are mutually a data object type or data object type relation and a task type.

Task type relations specify the data interfaces of the tasks (i.e. data consumed and produced). The context of each task is, thus defined by all of its incoming and outgoing data objects. Like the data perspective also this dynamic, process-oriented part of the model is steadily maturing during use.

In further discussion, we focus on the instance model, as we did with the data perspective before. Equation (4) provides the sets for the task perspective on instance level and for the overall instance graph G_I – data and task perspectives as follows:

$$T = \{\mathrm{t}_0, \ldots, \mathrm{t}_n\} \qquad Y^k = \{\mathrm{y}_0^k, \ldots, \mathrm{y}_n^k\} \qquad G_I = (T \cup D \cup Y \cup R) \tag{4}$$

As per definition, every data object is unique within the TEAM model, an integrated instance graph G_I is built via task relations, which further allows for traceability of the data and the task perspectives.

The data and task perspectives are integrated via task relations of different kinds. An example is given in Figure 3, considering data and tasks on instance level and a selected set of task relations of different kinds. We show a small part of the scenario described in Section 2, namely, when a user d°_{99} instantiates y^i_0 (i.e. y of kind i) the task t_0, allocate y^a_1 (i.e. y of kind a) and processes y^p_2 (i.e. y of kind p) the task to link the incoming e-mail d°_9 to the relevant case file d°_{11}. Incoming data objects are related via task relations of kind *reference (r)*, e.g. the task relations y^r_3 and y^r_4. To trace all activities on data, task relations are also used on data object relations. For example, the task relation y^c_5 indicates that the data object relation r°_{11} has been created (i.e. y of kind c) by the task t_0.

Table 2 provides an overview of the types of all instances involved in Figure 3.

Task relations also denote the exchange of data objects between tasks, thereby describing the underlying processes and making communication between users explicit. Thus, the orientation of the corresponding task type relations explicitly documents the process direction.

As data objects are unique in the instance model, they can be used to identify traces. For example, a data object, which was created by a first task and then further used in a subsequent one, is linked to both tasks via task relations. Thus, an entire trace between a start and an end node can be described. An example for such process traces is given in Figure 4. The set of all paths between two tasks is created following all outgoing task relations of kind creates y^c_i and ingoing y^r_i of kind references. To get a description in terms of tasks only, the number of paths can be reduced by eliminating the interim data objects, as subsequent tasks are typically interconnected by more than one data object.

As either tasks or data objects can be chosen as start and end points for these traces, not only process traces but also data life-cycles can be identified.

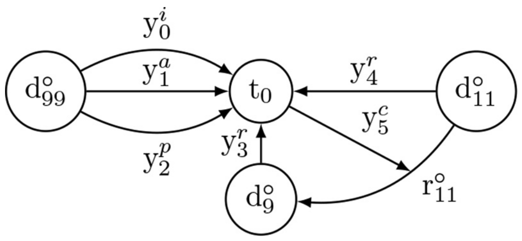


Figure 3. Graph G_I, integrating the data and task perspectives

Id	Type
t_0	ProcessEMail
d°_{99}	User
d°_9	EMail
d°_{11}	CaseFile
r°_{11}	ActHasEMail of the kind has
y^i_0	TTR.User-ProcessEMail.instantiates
y^a_1	TTR.User-ProcessEMail.allocates
y^p_2	TTR.User-ProcessEMail.processes
y^r_3	TTR.ProcessEMail-EMail.references
y^r_4	TTR.ProcessEMail-CaseFile.references
y^c_5	TTR.CaseFile-EMail.creates

Table 2. Details for the instances in Figure 3

5. The TE𝔸M model and knowledge graphs

When studying the relevant literature, it becomes obvious that there is no single, commonly accepted understanding of what constitutes a *knowledge graph*. Bergman (2019) collected over 25 different definitions, going back to the 1970s and showing a significant spike in citations (due to Google's knowledge graph) from 2012 up to now. Considering all diverse definitions, Bergman (2019) comes up with a free as possible definition, i.e. *a representation of knowledge (however defined) in the structural form of a directed (mostly acyclic) graph*. As this definition leaves an open space for great diversity in representation, application and construction (Ji *et al.*, 2020), we follow the recommendation of Hogan *et al.* (2019) and highlight our view on knowledge graphs in the context of the TE𝔸M model.

In general, we see knowledge graphs as follows:

5.1 Knowledge bases of types, instances and their attributes

[similar to the definitions in Paulheim (2017), d'Amato *et al.* (2019), Hogan *et al.* (2019) and Paulheim (2017)]. As the early evolution of the semantic web, there have been tensions between formalism and flexibility in knowledge representation. On the one hand, there is a need for a formal representation (e.g. in an ontology written in Web Ontology Language (OWL)) or at least for formal conceptual schemas and on the other hand, there is a tendency toward diversity and decentralized approaches to vocabularies (Bergman, 2019).

Although knowledge graphs, in general, are rather data-focused (i.e. intuitive) than schema-focused (i.e. descriptive) Polleres (2019), within the TE𝔸M model we need a *flexible* schema. We cannot use a strict ontology because we want the model to evolve dynamically at run-time (driven by the user) and to integrate heterogeneous sources. However, complete freedom of action is not applicable in many areas, e.g. the legal domain, where apart from ensuring certain data quality, legal requirements must be fulfilled and decisions need to be traceable at any time.

5.2 Guarded by constraints

We, therefore, defined *types* that represent templates used so far to guard the instances in the TE𝔸M model. Additionally, basic and domain-specific constraints are defined in the *meta model* and the *domain model*.

5.3 User-driven construction

We start with a small set of trustworthy types and then let users dynamically extend the model. The key idea is allowing the users to implicitly "build" types at run-time by constructing instances. Afterward, it is considered what should be added to the schema for learning purposes.

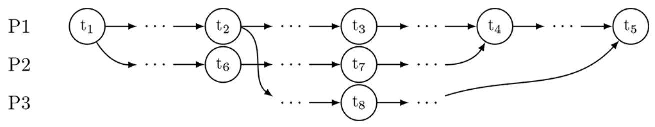


Figure 4.
Identifying paths P1, P2 and P3 between two tasks t_1 and t_5, showing only task relation y_i^c, y_i^r and indicating sets of interim data objects with "..."

5.4 Organized in a logical and computable graph

Although there are several works that suggest resource description framework (RDF) (and OWL) as a solid basis for knowledge graphs (Polleres, 2019; Ehrlinger and Wöß, 2016; Villazon-Terrazas *et al.*, 2017; Kejriwal, 2019), RDF is rarely used in classical information systems for domain-specific applications. We do not base our knowledge representation on RDF triples for the following reasons: The semantic web depends on complex standards of the semantic web stack (Table 3), whereas we go for reduced complexity and more flexibility in the TE𝔸M model. We further require a higher-level abstraction using an entity-relation conceptual schema for modeling and querying data (then it is granted for the lower level RDF data model) to prevent information overload for human users.

5.5 Integrated with a dynamic context

Knowledge graphs are designed rather statically, i.e. typically no temporal or actional context is included, which is also proved by the definitions in (Bergman, 2019). In the TE𝔸M model, knowledge objects are also linked, however, not only in a static sense (e.g. a natural person has an email address) but also through experience and use. We, thus enrich the knowledge graph with *process* semantics to record dynamic aspects such as data consumption/production of tasks, temporal validity of data and provenance (Polleres, 2019). Special attention is paid to *traceability* – to be able to explain at any time *why* objects are related to each other. Therefore, edges also have properties, such as timestamps, status information, etc., which can be the target of another edge, thereby creating, referencing or invalidating the relationship. This requires extending the classical understanding of graphs as nodes connected by edges by also allowing "edges connected by edges" using a kind of hyper-relation. Note that we do not use hypergraphs but rather an artificial node of type "edge," which is simply represented as an edge only. Thereby, the incoming/outgoing edges remain the same and our extended view on graphs can be mapped to RDF (via reification). Thus, equivalence to a classical graph is ensured.

5.6 Implemented in a multi-model graph database

For storing and managing the integrated knowledge and process graph, we deliberately chose a multi-model database, where an edge has the same base type as a node (i.e. document) and connects elements of the more general type document, to allow "edges-on-edges" in a simple and straightforward way.

Besides presenting our TE𝔸M model in terms of a knowledge graph, we now want to align it to the very well-known classification in the area of knowledge and KM, i.e. the Data Information Knowledge Wisdom (DIKW) pyramid, whose terms are still used and up-to-date (Zeleny, 1987; Ackoff, 1989; Akerkar and Sajja, 2009; Gajzler, 2016; Jennex, 2017; Frické, 2019).

Table 4 compares and maps the terms of the DIKW pyramid to the components of the TE𝔸M model.

Semantic web	TE𝔸M model
RDF triples (subject-predicate-object)	Instance model (entities, relations, attributes on both of them)
RDFS (RDF + schema)	Domain model (types, concepts, hierarchies, constraints)
OWL (RDFS + descriptive logic)	No direct mapping allows similar constraint definition in the meta model and domain model, graph-based knowledge representation
SPARQL	Path query language (Gremlin-like syntax)

Table 3. Comparing the TE𝔸M model to semantic web standards

Table 4.
Mapping the TE𝔸M model to DIKW terms

Terms	Ackoff (1989)	Zeleny (1987)	Akerkar and Sajja (2009)	TE𝔸M model
Data	Symbols	Know nothing	Raw observations	Observable data objects
Information	Data being processed	Know what	Aggregated, collected data	+ Non-observable data object(s) (types), data object (type) relations
Knowledge	Application of data/information, conveyed by instructions	Know how how-to	Human understanding, acquired by study and experience	+ Task(s) (types), task (type) relations
Understanding	Appreciation of *why*, conveyed by explanations	na	na	+ Task (type) relations on data object (type) relations
Wisdom	Evaluated understanding	Know why	Higher level of knowledge	Predictions and recommendations

Source: Extended from Jäger (2019)

Data, represented by symbols or static values, are expressed as observable data objects in the TE𝔸M model. Given a context, data become *information*, which represents processed, aggregated and collected data. In the TE𝔸M model, information is modeled as a non-observable data object(s) (types). Data object (type) relations aggregate non-observable data object(s) (types) to higher-level concepts and link them also to observable data objects at the finest level of granularity. Information, given meaning in a further step, becomes *knowledge*. Knowledge is obtained by (human) actors linking information and data through experience and transactions (Jäger *et al.*, 2016). In the TE𝔸M model, knowledge is made up by integrating data with tasks, i.e. *how-to* is expressed by sequences of the data object(s) (types) and task(s) (types) connected via task (type) relations.

In some domains, *understanding* is introduced as an additional level of an extended DIKW pyramid model between knowledge and wisdom. This appreciation of *why* is mapped to task (type) relations between task(s) (types) and data object (type) relations in the TE𝔸M model to support traceability. Further works also provide some extensions to the terms of Zeleny (1987), such as *know-who* (Meyer, 2010), which can naturally be expressed in the TE𝔸M model.

From knowledge/understanding, it is possible to gain insights leading to evaluated understanding (Ackoff, 1989) and actionable intelligence (Jennex, 2017). Thus, knowledge is information" sufficiently believed to be acted upon" (Bergman, 2018), which can be used for predictions and improvements within the TE𝔸M model through learning and mining methods.

In the following, concrete examples are provided based on our running example (Figures 2 and 3). While data, information, knowledge and understanding are used to conclude from the past, wisdom rather refers to the future (Jäger, 2019), which gives rise for predicting and proposing, for example, the best next task to a user:

data = [*doe@polymind.gmbh*] (d_2^*)

information = [*doe@polymind.gmbh* is an *EMailAddress* used in an *email*] ($d_2^* \leftarrow r_2^* - d_2^\circ \leftarrow r_3^\circ - d_4^\circ \leftarrow r_4^\circ - d_5^\circ \leftarrow r_5^\circ - d_9^\circ$)

knowledge = [*doe@polymind.gmbh* is needed for task *ProcessEMail*] ($d_9^\circ - \mathrm{y}_3^r \rightarrow \mathrm{t}_0$)

understanding = [*doe@polymind.gmbh* is related to a *CaseFile* through task *ProcessEMail*] $(d^{\circ}_{11} - r^{\circ}_{11} \rightarrow d^{\circ}_{9}, \mathrm{t}_0 - \mathrm{y}^{c}_{5} \rightarrow r^{\circ}_{11})$

6. User interaction with a dynamic, graph-based knowledge representation

Knowledge cannot be transferred directly between people; it requires a sender-receiver model, whereby the sender externalizes its knowledge and the receiver integrates it into her body of knowledge (Burkhard, 2005; Meyer, 2010).

Knowledge communication, therefore, plays a major role, especially in the areas of KM, communication studies and decision-making. Knowledge communication is defined as an activity of interactively conveying and co-constructing insights, assessments, experiences or skills through verbal and non-verbal means, which includes the successful transfer of know-how, know-why, know-what and know-who. Thereby, the advantage of computer-based visualization lies in the fact that it allows access, control, explore, combine and manipulate different types of complex data, information and data (Eppler, 2004; Burkhard, 2005; Meyer, 2010).

For managing knowledge and information, digital concept maps have been proposed (Tergan, 2005; Tergan *et al.*, 2006), which are intended to represent meaningful relationships between concepts in the form of propositions. Although it seems obvious to apply concept map representation to knowledge graphs, it becomes apparent that the complexity of such graphs quickly exceeds the capabilities of a visual representation. This applies in particular to the proposed TE𝔸M model, which, due to its multi-dimensional nature and the associated complexity, goes beyond a usual presentation frame, even for the small scenario shown in Figure 5, showing the instance graph of the proposed TE𝔸M model after the execution of 38 common tasks during the initial phase of patent drafting. It is, therefore,

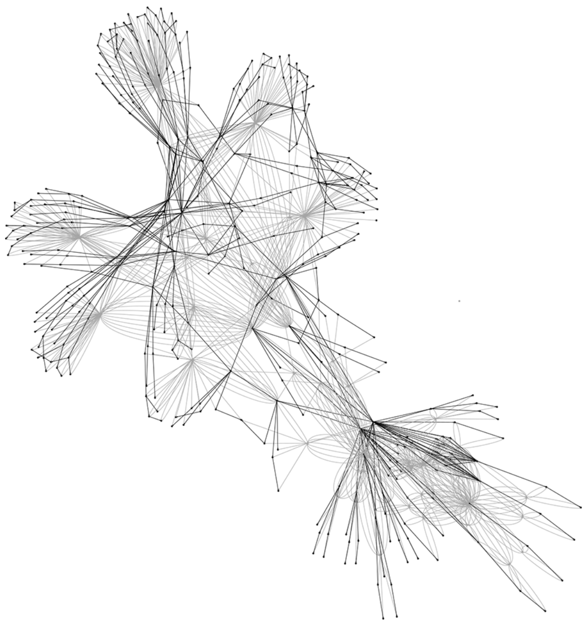


Figure 5. Resulting instance graph for a small real-world scenario with data objects and data object relations in black, as well as tasks and task in gray color

necessary to simplify the complexity of the graph for the user, but in particular to select suitable sections of the graph for user interaction.

Capturing the data objects required by a user to execute a task (that is relating those data objects to the task via task relations) creates a work context (*know-what*) that maps all work materials used. Therefore, a task contains a rationale for the changes made (*know-why*). Observing the accessed data objects also facilitates the externalization of the users' knowledge on what information is required to perform a certain task. Due to the fine-grained actions in a task (creates, validates, invalidates) at the level of data objects and data object relations, the detailed working approach of individual users becomes visible (*know-how*), not only within a task but also across several tasks. Finally, special task relations to data objects representing users were introduced to control access to tasks (know-who).

In the following, several key aspects of the user interface design are illustrated.

6.1 Representation of data objects

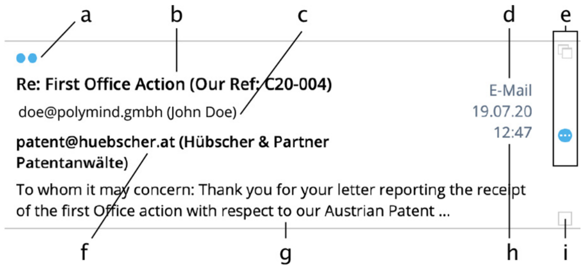
A first problem that arises in the visual representation of the proposed model is that only observable data objects have a value assigned and can, therefore, be directly represented. The displayable content of the majority of data objects results from their respective subordinate data objects, which together make up the visually presentable content of the superordinate data object. For example, after parsing, an *e-mail* is represented by a data object that has subordinate data objects such as *sender*, *receiver*, *subject* or *dateReceived*.

Thus, the visual representation of data objects is not possible without aggregating groups of data objects and, thereby, reducing graph complexity.

For a particularly simple representation of data objects, each corresponding data object type can have a string template assigned that, once rendered, allows a textual representation of the relevant data objects based on their related data objects. For example, a template for displaying an *e-mail* data object via the related *dateReceived*, *sender* and *subject* can be formulated as follows:

$$\$\{\text{dateReceived?datetime}\,\$\text{subject}\,(\text{from}:\ \$\text{sender})$$

Simple text representation requires sequential visual processing. However, to get a fast impression of the contents of a data object, a representation that is easily recognizable by pre-attentive visual processing (Treisman, 1985) is preferred. Especially, as the lack of pre-attentive visual processing is one of the common draw-backs of nowadays document and data management systems. Therefore, and as shown in Figure 6 in a block view, a data object component is introduced that allows for different visual representations of a data object.



Re: First Office Action (Our Ref: C20-004)
doe@polymind.gmbh (John Doe)
patent@huebscher.at (Hübscher & Partner Patentanwälte)
To whom it may concern: Thank you for your letter reporting the receipt of the first Office action with respect to our Austrian Patent …

E-Mail
19.07.20
12:47

Figure 6.
E-mail data object shown in block view

As can be seen in Figure 6, subordinate data objects can be displayed via nested data object components, like a status indicator as follows: (a) for displaying the presence of a related *dateProcessed* data object, a simple text representation for the *subject* (b), the *sender* (c), *receiver* (f), *textBody* (g) and *dateReceived* (h). In addition, the data object type name (d) can be displayed together with action buttons (e, i) like a button (i) to pick the data object. The nested structure allows for a fine-grained loading and updating of the user interface.

For a full visual representation of a data object, its user interface component supports a detail view, which is shown as part (b) of Figure 7. Preferably, a list of data objects (a) acts as an entry point for a detail view arranged next to the list, whereby multiple data objects can be selected and are displayed next to each other.

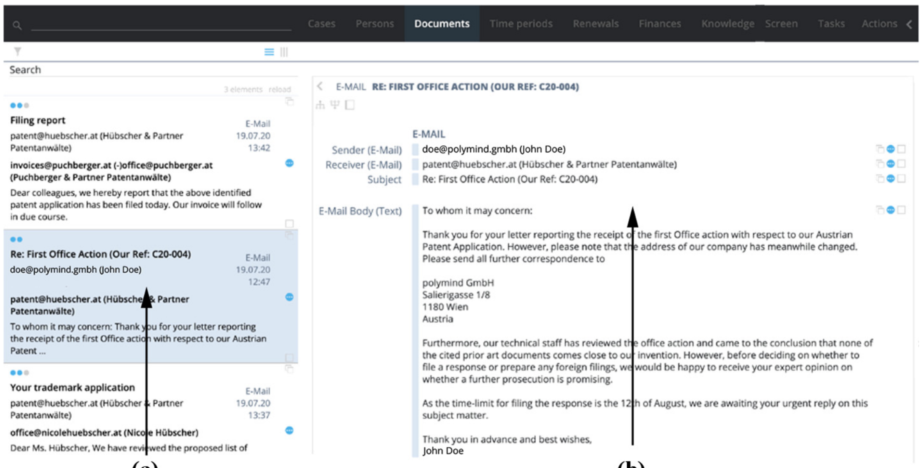
Furthermore, for each data object in the detail view, it is possible to switch between a predefined standard view and the data object's subordinates and superordinate's-related via data object relations. This also shows that abstract data objects with no assigned value are only abstract mental concepts that can only be visually represented and understood through the data objects linked to them.

Apart from a general default layout for data objects, the content of the different views can be overridden by assigning a specific renderer to the corresponding data object type. However, the pre-set default layout supports an evolving model, where the user can also add data object types and data object type relations as described in more detail in Section 6.5.

6.2 Searching for data objects

Similar to the representation of data objects, the search for data objects has to be based on observable data objects which can, therefore, be searched for directly. To search for more complex data objects, a graph traversal of predefined length against the direction of neighboring data object relations is performed for each resulting data object with a matching value. For the resulting paths, matching data object nodes are identified and returned as the search result.

Continuing the example above, if a user enters the search term John polymind, the search term is first split into the separate tokens John and polymind, for both of which a search for data objects with a matching value is performed, resulting in the start nodes John Doe and



Filing report
patent@huebscher.at (Hübscher & Partner Patentanwälte)
E-Mail
19.07.20
13:42
invoices@puchberger.at (-)office@puchberger.at (Puchberger & Partner Patentanwälte)
Dear colleagues, we hereby report that the above identified patent application has been filed today. Our invoice will follow in due course.

Re: First Office Action (Our Ref: C20-004)
doe@polymind.gmbh (John Doe)
E-Mail
19.07.20
12:47
patent@huebscher.at (Hübscher & Partner Patentanwälte)
To whom it may concern: Thank you for your letter reporting the receipt of the first Office action with respect to our Austrian Patent ...

Your trademark application
patent@huebscher.at (Hübscher & Partner Patentanwälte)
E-Mail
19.07.20
13:37
office@nicolehuebscher.at (Nicole Hübscher)
Dear Ms. Hübscher, We have reviewed the proposed list of

E-MAIL RE: FIRST OFFICE ACTION (OUR REF: C20-004)

E-MAIL
Sender (E-Mail) doe@polymind.gmbh (John Doe)
Receiver (E-Mail) patent@huebscher.at (Hübscher & Partner Patentanwälte)
Subject Re: First Office Action (Our Ref: C20-004)

E-Mail Body (Text) To whom it may concern:

Thank you for your letter reporting the receipt of the first Office action with respect to our Austrian Patent Application. However, please note that the address of our company has meanwhile changed. Please send all further correspondence to

polymind GmbH
Salierigasse 1/8
1180 Wien
Austria

Furthermore, our technical staff has reviewed the office action and came to the conclusion that none of the cited prior art documents comes close to our invention. However, before deciding on whether to file a response or prepare any foreign filings, we would be happy to receive your expert opinion on whether a further prosecution is promising.

As the time-limit for filing the response is the 12th of August, we are awaiting your urgent reply on this subject matter.

Thank you in advance and best wishes,
John Doe

Figure 7.
E-mail data object of Figure 6(a) shown in detail view (b)

doe@polymind.gmbh. Traversing from these two nodes upwards against the direction of data object relations, the first matching node would be the *eMailContact* of John Doe and further *naturalPerson* John Doe. Both would be returned as search results.

6.3 Enforcing the explicit selection of pre-requisite data objects

Another specific problem in this context is that in contrast to conventional user interfaces, not only the data objects entered but also the data objects requested by the user should be captured.

To enforce explicit referencing of data objects required to perform a certain task, the user interface is split into areas for read-only presentation, search and navigation of data objects and random access of data objects in the context of tasks.

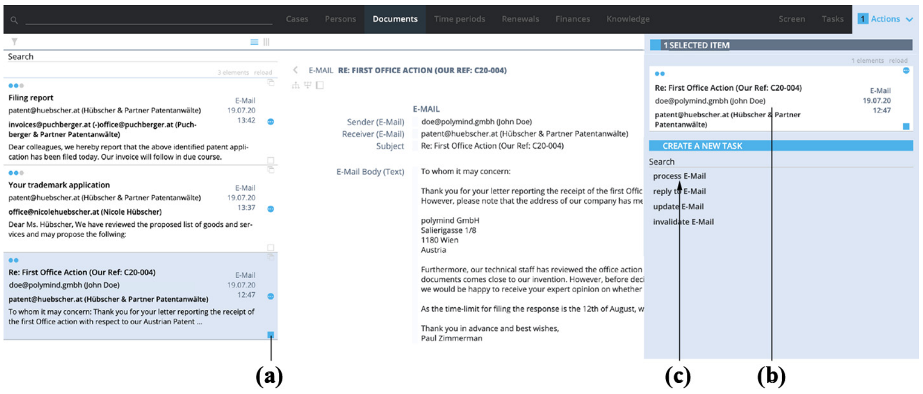
As an interface between those two areas, the well-known metaphor of a shopping cart was introduced as follows: Throughout the whole user interface, data objects can be picked and collected into a cart. Depending on the data objects within the cart, corresponding task types can be selected via an *action*-menu as shown in Figure 8. The selectable task types are determined by searching the domain model for those task types, that have task type relations to all data object types of the data objects within the cart.

Continuing the above example, for processing the incoming *e-mail*, a first administrative user can pick the *e-mail* data object and open the action menu on the top right. Consequently, a list of possible task types is shown, i.e. such task types, which have at least one task type relation to the data object type of the picked *e-mail* data object. It should be emphasized that the user is not restricted to select the complete *e-mail* data object but can also pick subordinate data objects thereof as required.

6.4 Enforcing unique data objects within the system

One of the key concepts of the TEAM model is the uniqueness of data objects for relating tasks with each other via task relations to common data objects. This is a prerequisite to form interconnected process traces within the graph.

While the reusability of more complex data objects is generally seen as an advantage, it is difficult to explain to users why simple data object types, such as *string*, *date* or other primitive data types, should not be entered traditionally but need to be selected or created



Filing report
patent@huebscher.at (Hübscher & Partner Patentanwälte)
invoices@puchberger.at (-)office@puchberger.at (Puchberger & Partner Patentanwälte)
Dear colleagues, we hereby report that the above identified patent application has been filed today. Our invoice will follow in due course.
E-Mail 19.07.20 13:42

Your trademark application
patent@huebscher.at (Hübscher & Partner Patentanwälte)
office@nicolehuebscher.at (Nicole Hübscher)
Dear Ms. Hübscher, We have reviewed the proposed list of goods and services and may propose the follwing:
E-Mail 19.07.20 13:37

Re: First Office Action (Our Ref: C20-004)
doe@polymind.gmbh (John Doe)
patent@huebscher.at (Hübscher & Partner Patentanwälte)
To whom it may concern: Thank you for your letter reporting the receipt of the first Office action with respect to our Austrian Patent ...
E-Mail 19.07.20 12:47

E-MAIL **RE: FIRST OFFICE ACTION (OUR REF: C20-004)**

E-MAIL
Sender (E-Mail): doe@polymind.gmbh (John Doe)
Receiver (E-Mail): patent@huebscher.at (Hübscher & Partner Patentanwälte)
Subject: Re: First Office Action (Our Ref: C20-004)

E-Mail Body (Text):
To whom it may concern:

Thank you for your letter reporting the receipt of the first Offic
However, please note that the address of our company has me

polymind GmbH
Salierigasse 1/8
1180 Wien
Austria

Furthermore, our technical staff has reviewed the office action
documents comes close to our invention. However, before deci
we would be happy to receive your expert opinion on whether

As the time-limit for filing the response is the 12th of August, w

Thank you in advance and best wishes,
Paul Zimmerman

Re: First Office Action (Our Ref: C20-004)
doe@polymind.gmbh (John Doe)
patent@huebscher.at (Hübscher & Partner Patentanwälte)
E-Mail 19.07.20 12:47

process E-Mail
reply to E-Mail
update E-Mail
invalidate E-Mail

Notes: (a) Into a basket; (b) and selecting a task type for instantiating a task

Figure 8.
Picking one or more data object(s) with the button

explicitly. To overcome this problem, the hierarchy levels of the least expressive data objects were usually hidden and replaced by generally known input boxes.

However, when a data value is entered into these input boxes, the TE𝔸M system first searches for a data object with the respective value before creating a new one.

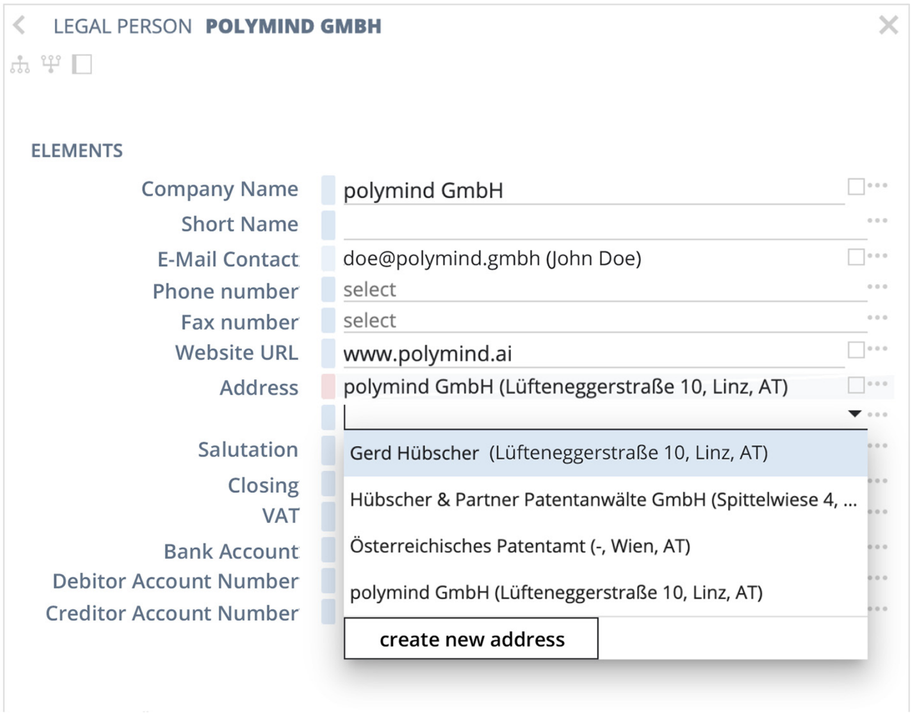
With more complex data objects, the concept of reusable, that means pick-, drag- and drop-able, building blocks were pursued to support playful navigation and composition of related data objects.

As shown in Figure 9, an administrative user, who changes the correspondence *address* of the patent proprietor first invalidates the data object relation between the *legalPerson* and the old *address* with the option to invalidate the *address* data object as well in case the old *address* will or should not be reused. Afterward, she adds a new *address* data object by opening up the editor, first searching for the new *address* and creating one in case no results are found.

6.5 Execution of tasks and model evolution

Modifying data objects by creating, removing, validating or invalidating data object relations is, according to the definition of the TE𝔸M model, exclusively performed within the scope of tasks. As shown in Figure 10, each possible modification action creates a corresponding task relation, data modification can be tracked at a fine-grained level. As each

composition of related data objects.

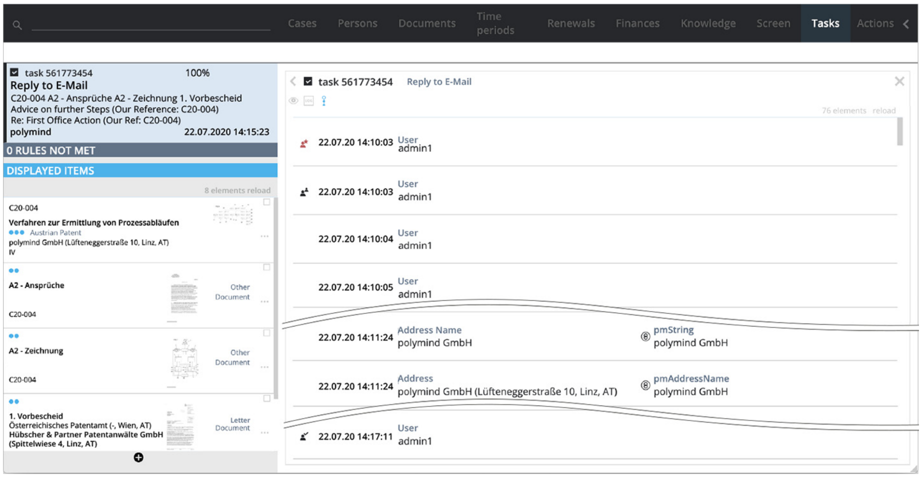

LEGAL PERSON **POLYMIND GMBH**

ELEMENTS

Company Name	polymind GmbH
Short Name	
E-Mail Contact	doe@polymind.gmbh (John Doe)
Phone number	select
Fax number	select
Website URL	www.polymind.ai
Address	polymind GmbH (Lüfteneggerstraße 10, Linz, AT)
Salutation	
Closing	
VAT	
Bank Account	
Debitor Account Number	
Creditor Account Number	

Gerd Hübscher (Lüfteneggerstraße 10, Linz, AT)
Hübscher & Partner Patentanwälte GmbH (Spittelwiese 4, ...
Österreichisches Patentamt (-, Wien, AT)
polymind GmbH (Lüfteneggerstraße 10, Linz, AT)
create new address

Figure 9. Searching for new *address* data objects after invalidating the data object relation to the old *address* data object



task 561773454 100%
Reply to E-Mail
C20-004 A2 - Ansprüche A2 - Zeichnung 1. Vorbescheid
Advice on further Steps (Our Reference: C20-004)
Re: First Office Action (Our Ref: C20-004)
polymind 22.07.2020 14:15:23

0 RULES NOT MET

DISPLAYED ITEMS

8 elements reload

C20-004
Verfahren zur Ermittlung von Prozessabläufen
Austrian Patent
polymind GmbH (Lüfteneggerstraße 10, Linz, AT)
IV

A2 - Ansprüche
C20-004
Other Document

A2 - Zeichnung
C20-004
Other Document

1. Vorbescheid
Österreichisches Patentamt (-, Wien, AT)
Hübscher & Partner Patentanwälte GmbH
(Spittelwiese 4, Linz, AT)
Letter Document

task 561773454 Reply to E-Mail

76 elements reload

```
22.07.20 14:10:03  User
                   admin1
22.07.20 14:10:03  User
                   admin1
22.07.20 14:10:04  User
                   admin1
22.07.20 14:10:05  User
                   admin1
22.07.20 14:11:24  Address Name                                  pmString
                   polymind GmbH                                 polymind GmbH
22.07.20 14:11:24  Address                                       pmAddressName
                   polymind GmbH (Lüfteneggerstraße 10, Linz, AT)  polymind GmbH
22.07.20 14:17:11  User
                   admin1
```

Figure 10.
Task history is based on all task relations to data objects and data object relations and gives a complete overview of the actions performed

task relation is reflected by a task type relation in the domain model, the way of performing a task can be modeled beyond instance level and allows the support of users, who require assistance on how to perform certain task types. However, knowledge workers with the appropriate access rights can expand the domain model by simply adding task relations of new task type relations, data object relations of new data object type relations or even by adding or modifying data objects and tasks of new data object types and task types.

In the aforementioned example, the administrative user could add a new data object type *employee*, relate it to *legalPerson* and add John Doe to the *legalPerson* polymind GmbH.

6.6 Context-based knowledge work

The introduction of tasks connected to data objects via task relations makes it difficult to decide which data objects are to be displayed within the tasks as an entry point for further navigation. For example, when assigning an *e-mail* to a *case file*, from a hierarchical perspective, showing the *case file* as an entry point would be sufficient. However, that way, the main aspects of the task performed remain hidden from the user. On the other hand, displaying all modifications that occurred within a task by showing all task relations to the user is not suitable either. Therefore, we decided to reuse the information learned from the prescribed method of integrating a random-access task view with the possibility of picking data objects throughout the system to allow for a context-based view of data objects required to perform a task as follows: data objects explicitly selected by the user to perform a task are related to this task via a separate task relation of kind *displays*, thereby modeling what data objects matter to the user within the context of the task. Such an explicit selection occurs during the initial picking of data objects required to execute the task but can be altered and modified later on, too.

Figure 11 shows a two-screen view of the random-access task area, allowing the knowledge worker to arrange data objects as she prefers. At the same time, the user can view data objects and create new ones, for example, in the above example, to write a response e-mail taking into account all relevant data objects within the context of the current task.

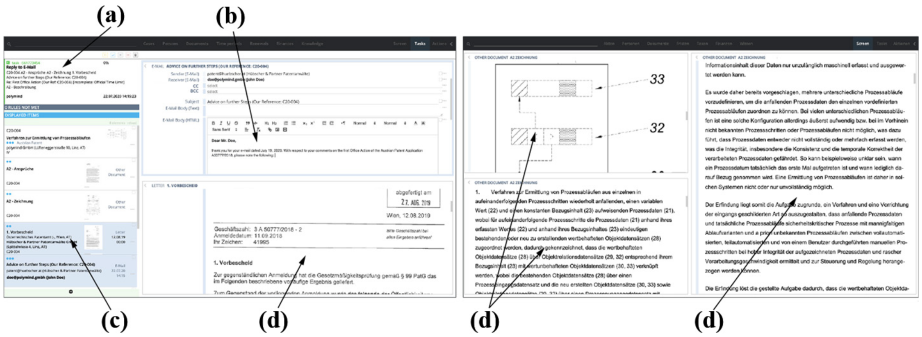


Figure 11. Knowledge work: within a task

Notes: (a) A user can display relevant data objects; (c) and arrange them in a detail view; (d) as required. Furthermore, data objects can also be edited (b)r

6.7 Delegation of tasks as a traceable alternative to other ways of communication

One problem of nowadays e-mail communication is that apart from using suitable file formats, information is usually not exchanged in a structured, reusable way. In the legal domain, tasks are delegated and assigned usually via deadlines, physical file handling or subsidiary via e-mail communication.

Although within the TE^{A}M system, e-mails can be processed as data objects, information can also be shared by creating a task referencing the required data objects and assigning this task to another user. Thereby, the unique data objects are linking tasks performed by different users via task relations, thus building process traces.
For example, an administrative user can pass the task of substantially replying to an e-mail to a knowledge worker by referencing the *e-mail* data object and any further information such as *client*, *case file* and assign the task to that knowledge worker.

This results in a short setup time due to the pre-compiled set of referenced data objects. However, task delegation requires that the task creator is aware of which data objects are required for the subsequent task. Following the intention of the TE^{A}M model, those prerequisite data objects are identified over time and are available to other users via the domain model. As a transitional solution, the TE^{A}M model still allows for the creation of intermediate data objects like time limits, which can be related to tasks by the executing user.

6.8 Tracing knowledge work and switching between the graph dimensions

Once data object uniqueness is assured, each creation and usage of a data object creates links (task relations) between two or more tasks. Therefore, the source task of every data object, as well as every usage of the data object can be traced and recalled. As tasks relate to all other data objects used and created while creating or using that data object, the respective working context can be restored.

However, the representation of the process dimension in an ordinary user interface environment is difficult because it has to be integrated into a static view of the data objects and their data object relations to each other, providing information about their creation, consumption and modification over time.

Therefore, for each data object, a separate process popup view is developed, which allows the creating and consuming tasks to be displayed and navigated to (Figure 12).

Displaying a trace of several consecutive tasks fails without further measures due to the lack of a defined start and end point. Thus, starting from a data object, the model graph expands in both process directions in the form of one or more trees, which already have an enormous and no longer visually presentable dimension after traversing a few edges and nodes in the emerging graph.

For enabling initial navigation through the emerging process traces, the user is, therefore, enabled to select one subsequent or preceding task step by step from the process popup view, and thus query a concrete task sequence based on a starting data object.

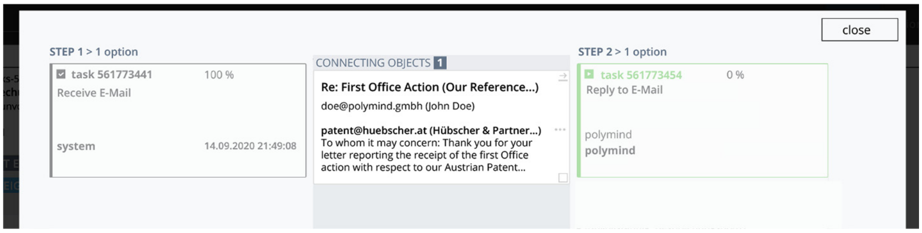
Such a task sequence is shown in Figure 13, revealing the amount of connecting data objects between two tasks and allowing the user to expand the view analogous to Figure 12 for displaying the interconnecting data objects between the relevant tasks.

7. Results

The TE^{A}M system prototype was evaluated in two case studies with Austrian patent law firms. User feedback was collected through open interviews with the administrative staff and knowledge workers who participated in the case studies.

7.1 User perception

After working on the use cases, both user groups reported satisfying experiences concerning the presentation of information within the test system. However, it took some getting used to having to explicitly perform a task to change data objects. In particular, it was observed that many users had difficulties in selecting the appropriate task type or in specifying a new task type correctly before actually executing the planned activity. To change multiple data objects regardless of their origin or context within one task was a new experience for the users and enabled them to focus on the task in its context while requiring less user interaction. However, due to the possibility to change existing task types and to edit all data objects within a task, the users tended to extend existing general task types instead of creating new specific ones.

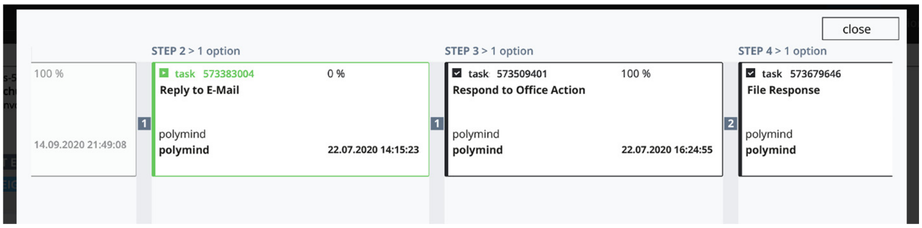


task 561773441 100 %
Receive E-Mail
system 14.09.2020 21:49:08

Re: First Office Action (Our Reference...)
doe@polymind.gmbh (John Doe)
patent@huebscher.at (Hübscher & Partner...)
To whom it may concern: Thank you for your letter reporting the receipt of the first Office action with respect to our Austrian Patent...

task 561773454 0 %
Reply to E-Mail
polymind
polymind

Figure 12.
Creating and consuming tasks for a data object



100 %
14.09.2020 21:49:08

task 573383004 0 %
Reply to E-Mail
polymind
polymind 22.07.2020 14:15:23

task 573509401 100 %
Respond to Office Action
polymind
polymind 22.07.2020 16:24:55

task 573679646
File Response
polymind
polymind

Figure 13.
Navigating through the traced process

On the other hand, changing or specifying new data object types was intuitive for most knowledge workers and the administrative staff within their own knowledge domains. Defining types for both groups of users, which is needed when data objects are handed over by tasks, requires more communication between the affected groups. The tests performed showed that users tend to use more general task types and data object types in favor of more specific ones. One reason, therefore, might be the extended effort to define types and to choose whether to model aspects as task types or as data object types. For example, one can model as follows: specific document types like *eMail* or *letter*, all referenced by the same more general task type *sendDocument* or specific task types like *sendEMail* or *sendLetter*, all referencing a more general data object type *document*.

7.2 Technical aspects

The TE^{A}M system prototype shows that the integration of the knowledge and process perspectives leads to a very complex graph model, especially with the process perspective, as every change of a data object or a data object relation results in at least one task relation.

Despite the proposed TE^{A}M system model having a limited set of core components, the model graph derived from a real-world scenario (the user interactions of four users during the application draft and initial prosecution phase of the Austrian patent AT521649) exceeds an easily ascertainable level of complexity. For example, a relatively small data set of 17 (t_0), 27 (t_2) and 38 (t_4) common tasks during the initial phase of patent drafting lead to the graphs shown in Figure 14.

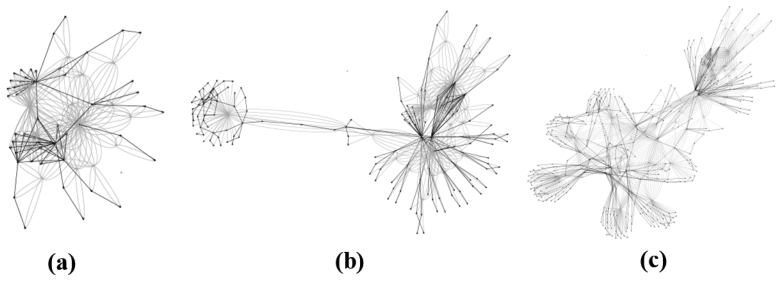
For the evaluation of the first testing results, we introduce the following metrics to describe graph characteristics at a certain point in time as follows:

Besides the cardinality of the sets Δ, P, Ξ, Υ, D, D^*, R, T and T (Table 5), we define the relation between the number of observable data objects $|D^*|$ and the overall number of data objects $|D|$ as observability $m_{observe}$ of the mental concepts within the graph G_I.

$$m_{observe} = \frac{|D^*|}{|D|} \tag{5}$$

This metric gives an indication of how well abstract, i.e. non-observable mental concepts can be encoded within the information graph G_I.

Furthermore, we observe whether the information representing the mental concepts is entailed within the data objects or the data object relations. Therefore, we introduce the



Notes: (a) after t_0; (b) after t_2; (c) after t_4

Figure 14. A part of the instance graph after three points in time, showing data objects and data object relations in black and tasks, as well as task relations in gray with the increasing scale from left (a) to right (c)

Table 5.
Overview of metrics collected during data set generation

Metric	t_0	t_1	t_2	t_3	t_4
Data object type count $\|\Delta\|$	208.000	208.000	209.000	212.000	214.000
Data object type relation count $\|P\|$	201.000	202.000	224.000	242.000	259.000
Task type count $\|\Xi\|$	131.000	131.000	131.000	135.000	139.000
Task type relation count $\|\Upsilon\|$	281.000	292.000	369.000	492.000	717.000
Data object count $\|D\|$	48.000	102.000	150.000	202.000	353.000
Observable data object count $\|D^*\|$	33.000	60.000	76.000	91.000	137.000
Data object relation count $\|R\|$	62.000	125.000	181.000	247.000	435.000
$m_{observe}$	0.687	0.588	0.506	0.450	0.388
m_{entail}	0.774	0.816	0.828	0.817	0.811
Task count $\|T\|$	17.000	26.000	27.000	31.000	38.000
Task relation count $\|Y\|$	418.000	618.000	731.000	933.000	1,479.000
m^N_{shape}	2.823	3.923	5.555	6.516	9.289
m^E_{shape}	0.148	0.202	0.247	0.264	0.294

relation m_{entail} between the total number of data objects $|D|$ and the total number of data object relations $|R|$ as follows:

$$m_{entail} = \frac{|D|}{|R|} \tag{6}$$

Considering data objects as symbols within a hierarchical structure of mental concepts, m_{entail} is also a measure for the completeness of the symbol set. In other words, we expect a saturation with respect to data objects as the model evolves because already existing data objects are reused over time.
Finally, we introduce metrics for the graph shape in terms of dimension with respect to data object components $|D|$, $|R|$ and task components $|T|$, $|P|$, separated in a node-related metric m^N_{shape} and an edge-related metric m^E_{shape}:

$$m^N_{shape} = \frac{|D|}{|T|} \qquad m^E_{shape} = \frac{|R|}{|P|} \tag{7}$$

The total number of data object types and task types in the initial configuration (t_0) is given by a preset model and remained relatively constant throughout the test, which indicates a nearly complete set of data object types and task types. However, the value progression of the total count of data object type relations and task type relations shows that the relations among data object types and task types were barely modeled initially, but completed during user interaction. Especially process-related aspects, namely, which data object types and data object type relations are created, validated or invalidated within which task type, were captured over time, thereby resulting in a comprehensive description of the tasks performed.

With respect to instances, we observe a significant decrease of $m_{observe}$ over time, which confirms that more and more information is captured via abstract, non-observable data objects. Although both data objects and data object relations increase over time, a sum of squares regression analysis shows a greater increase of data object relations over time.

In this regard, it should be noted that the entailment metric m_{entail} first increases from t_0 – t_2 before decreasing slightly from t_3 –t_4. This confirms our initial expectation for m_{entail} being dependent on the completeness of the symbol set spanned by data objects. While increasing during an initial phase of building up the symbol set, m_{entail} decreases again as existing data objects get reused and are only related to each other via new data object relations.

Finally, the shape metric m^N_{shape} reveals a much larger amount of data objects compared to tasks, that even increases over time, which confirms that tasks establish a procedural context for several data objects at once. The second shape metric m^E_{shape}, related to data object relations and task relations was already expected to be low, as task relations relate not only tasks and data objects but also tasks and data object relations. Furthermore, the task status is expressed as a group of task relations to data objects (representing users) as well. However, $m^E_{shape} << 1$ shows the importance of a task-centered modeling approach when it comes to capturing observable user behavior.

Nevertheless, the vast majority of task relations indicates a possible bottleneck in process-oriented graph traversals and can be seen as a hint to reduce and transfer parts of the information contained therein.

8. Conclusion

To overcome the shortcomings of current approaches and IT systems concerning the integration of knowledge and processes, as well as dynamically evolving mental concepts of communication-intensive application domains, we propose the TE^{A}M model.

A key research result is an approach to flexibly manage data, information, knowledge, tasks and processes within a mixed administrative and knowledge work domain. This approach has been evaluated with a prototypical implementation in real-world environments. The use cases heavily support our hypothesis that the TE^{A}M model is suitable to support knowledge and administrative work equally within one integrated model. This makes a fundamental difference to traditional BPM approaches with not only a focus on administrative work but also to more flexible and adaptable approaches for knowledge-intensive business processes or case handling.

The TE^{A}M model provides a meta model along with the dimensions as follows: knowledge in the form of the domain, organizational, operational knowledge and the dynamic behavior in the form of sequences of tasks and interconnecting data objects. The meta model specifies how to describe both, the domain model with domain-specific mental concepts (types) and their relations, as well as the instance model.

To allow for the required flexibility and adaptability, the domain model and the instance model are continuously evolving by user interaction. The instance model is specified at run-time based on information and communication flows in a bottom-up perspective, considering execution-specific data and the dynamic assignment of tasks.

Creating the multi-dimensional knowledge/process graph (e.g. inserting basic domain data, user and task types), we succeeded in designing a highly flexible and adaptable model, which supports the evolution of the domain model, the instance model and the integration of the static and dynamic aspects. The state-of-the-art approaches we discussed in the related work section do not support these characteristics by far.

Evaluating the prototype in a real-world setting showed that the developed user interface design is suitable to hide the complexity of the model from the users and still allows for flexibly extending and adapting the system not only on the instance but also on the type level.

Business processes are recorded through traces of tasks and data objects in the knowledge/process graph. In future work, we will use this fine-grained tracing to as follows: identify new data object types and task types based on a set of instances and for graph-based process mining on the sequences of tasks interconnected by data objects to discover recurrent (hidden) execution patterns.

Note

1. Despite of different definitions of the term *mental concepts*, we consider a mental concept as an abstract representation of a certain thing or set of things, such as people, objects, places or actions that can be organized in hierarchies.

References

Ackoff, R.L. (1989), "From data to wisdom", *Journal of Applied Systems Analysis*, Vol. 16 No. 1, pp. 3-9.

Ahsan, S. and Shah, A. (2006), "Data, information, knowledge, wisdom: a doubly linked chain", In *Proceedings of IKE*, Citeseer, pp. 270-278.

Akerkar, R. and Sajja, P. (2009), *Knowledge-Based Systems*, Jones and Bartlett Publishers.

Auer, D., Geist, V. and Draheim, D. (2009), "Extending BPMN with submit/response-style user interaction modeling", In *2009 IEEE Conference on Commerce and Enterprise Computing*, IEEE, pp. 368-374, doi: 10.1109/CEC.2009.75.

Bailey, D.E., Leonardi, P.M. and Chong, J. (2010), "Minding the gaps: understanding technology interdependence and coordination in knowledge work", *Organization Science*, Vol. 21 No. 3, pp. 713-730.

Bergman, M.K. (2018), "Information, knowledge, representation", In *A Knowledge Representation Practionary: Guidelines Based on Charles Sanders Peirce*, Springer, pp. 15-42, doi: 10.1007/978-3-319-98092-8_2.

Bergman, M.K. (2019), "Common sense view of knowledge graphs", available at https://www.mkbergman.com/2244/a-common-sense-view-of-knowledge-graphs (accessed 3 August 2021).

Burkhard, R.A. (2005), "Knowledge visualization: the use of complementary visual representations for the transfer of knowledge. A model, a framework, and four new approaches", PhD thesis. ETH Zurich.

d'Amato, C., Kirrane, S., Bonatti, P.A., Rudolph, S., Krötzsch, M., van Erp, M. and Zimmermann, A. (2019), "Foundations", in *Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Reports)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, pp. 79-89, Vol. 8 No. 9, available at: https://drops.dagstuhl.de/opus/volltexte/2019/10328/pdf/dagrep_v008_i009_p029_18371.pdf

Dalkir, K. (2005), *Knowledge Management in Theory and Practice*, 1st ed. Elsevier textbooks, Burlington and USA, available at: http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10226635.

Davies, J.F., Grobelnik, M. and Mladenic, D. (2009), *Semantic Knowledge Management: Integrating Ontology Management, Knowledge Discovery, and Human Language Technologies*, 1st ed. Springer Publishing Company.

Döhring, M., Zimmermann, B. and Karg, L. (2011), "Flexible workows at designand runtime using BPMN2 adaptation patterns", In *International Conference on Business Information Systems*, Springer, pp. 25-36.

Domingue, J., Fensel, D. and Hendler, J.A. (Eds) (2011), *Handbook of Semantic Web Technologies*, Springer Science and Business Media, Berlin Heidelberg.

Drucker, P.F. (1959), *Landmarks of Tomorrow: A Report on the New 'Post-Modern' World*, Harper and Brothers, New York, NY.

Drucker, P.F. (1999), "Knowledge-worker productivity: the biggest challenge", *California Management Review*, Vol. 41 No. 2, pp. 79-94.

Dumas, M., La Rosa, M., Mendling, J. and Reijers, H.A. (2013), *Fundamentals of Business Process Management*, Heidelberg: Springer Berlin Heidelberg, Berlin, doi: 10.1007/978-3-642-33143-5.

Ehrlinger, L. and Wöß, W. (2016), "Towards a definition of knowledge graphs", *SEMANTiCS (Posters, Demos, SuCCESS)*, Vol. 48, pp. 1-4.

Eppler, M.J. (2004), "Knowledge communication problems between experts and managers: an analysis of knowledge transfer in decision processes", Tech. rep., Università della Svizzera Italiana.

Firestone, J.M. and McElroy, M.W. (2012), *Key Issues in the New Knowledge Management*, Routledge.

Frické, M. (2019), “The knowledge pyramid: the DIKW hierarchy”, *Knowledge Organization*, Vol. 46 No. 1.

Gajzler, M. (2016), “Usefulness of mining methods in knowledge source analysis in the construction industry”, *Archives of Civil Engineering*, Vol. 62 No. 1, pp. 127-142.

Geist, V., Illibauer, C., Natschläger, C. and Hutter, R. (2016), “Supporting customizable business process models using graph transformation rules”, *International Journal of Information System Modeling and Design (IJISMD)*, Vol. 7 No. 3, pp. 51-71.

Gronau, N. and Weber, E. (2004), “Management of knowledge intensive business processes”, in Desel, J., Pernici, B. and Weske, M. (Eds), *International Conference on Business Process Management*, Lecture Notes in Computer Science, Springer Berlin Heidelberg, Vol. 3080, pp. 163-178, doi: 10.1007/978- 3- 540- 25970- 1_11.

Grünert, D., Brucker-Kley, E. and Keller, T. (2014), “OBPM–an opportunistic approach to business process modeling and execution”, *International Conference on Business Process Management*, Springer, pp. 463-474.

Hallerbach, A., Bauer, T. and Reichert, M. (2010), “Capturing variability in business process models: the provop approach”, *Journal of Software Maintenance and Evolution: Research and Practice*, Vol. 22 Nos 6/7, pp. 519-546.

Hepp, M. and Roman, D. (2007), “An ontology framework for semantic business process management”, In *Wirtschaftinformatik Proceedings 2007*, p. 27.

Hepp, M., Leymann, F., Domingue, J., Wahler, A. and Fensel, D. (2005), “Semantic business process management: a vision towards using semantic web services for business process management”, In *IEEE International Conference on e-Business Engineering (ICEBE’ 05)*, IEEE, pp. 535-540.

Hevner, A.R., March, S.T., Park, J. and Ram, S. (2004), “Design science in information systems research”, *MIS Quarterly*, Vol. 28 No. 1, pp. 75-105, available at: http://dl.acm.org/citation.cfm?id=2017212.2017217

Hogan, A., Brickley, D., Gutierrez, C., Polleres, A. and Zimmermann, A. (2019), “(Re)defining knowledge graphs”, in *Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Reports)*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Vol. 8 No. 9, pp. 74-79, available at: https://drops.dagstuhl.de/opus/volltexte/2019/10328/pdf/dagrep_v008_i009_p029_18371.pdf

Jäger, M. (2019), “Working with trust and precision of information and data in knowledge processing systems/eingereicht von dipl.-ing. Markus Jäger, BSc. MLBT”, PhD thesis. Universität Linz.

Jäger, M., Phan, T.N. and Nadschläger, S. (2016), “The trustworthiness of data in smart homes”, *FAIMA Business and Management Journal*, Vol. 4 No. 3, p. 17.

Jennex, M.E. (2017), “Big data, the internet of things, and the revised knowledge pyramid”, *ACM SIGMIS Database: The DATABASE for Advances in Information Systems*, Vol. 48 No. 4, pp. 69-79.

Ji, S., Pan, S., Cambria, E., Marttinen, P. and Philip, S.Y. (2020), “A survey on knowledge graphs: Representation, acquisition and applications”, In IEEE Transactions on Neural Networks and Learning Systems, *arXiv preprint arXiv:2002.00388*.

Jiménez-Ramírez, A., Weber, B., Barba, I. and Del Valle, C. (2015), “Generating optimized configurable business process models in scenarios subject to uncertainty”, *Information and Software Technology*, Vol. 57, pp. 571-594.

Johnson-Laird, P.N. (1980), “Mental models in cognitive science”, *Cognitive Science*, Vol. 4 No. 1, pp. 71-115.

Karagiannis, D. and Telesko, R. (2000), “The EU-Project PROMOTE: a processoriented approach for knowledge management”, In *Proceeding of the Third International Conference of Practical Aspects of Knowledge Management*, pp. 9-18.

Kejriwal, M. (2019), “What is a knowledge graph?”, *Domain-Specific Knowledge Graph Construction*, Springer, pp. 1-7.

Künzle, V. and Reichert, M. (2011), “PHILharmonicFlows: towards a framework for object-aware process management”, *Journal of Software Maintenance and Evolution: Research and Practice*, Vol. 23 No. 4, pp. 205-244, doi: 10.1002/smr.524, available at http://dbis.eprints.uni-ulm.de/714/1/JSME_Kuenzle_Reichert_11.pdf

Künzle, V. (2013), "Object-Aware process management", *Dissertation*, Universität Ulm, Ulm.

McElroy, M.W. (2003), *The New Knowledge Management: Complexity, Learning, and Sustainable Innovation*, Butterworth-Heinemann, Amsterdam [u.a.].

Marin, M., Hull, R. and Vaculin, R. (2013), "Data centric BPM and the emerging case management standard: a short survey", in La Rosa, M., Soffer, P. and LNBIP (Eds), *BPM 2012 Workshops*, Springer-Verlag, Berlin Heidelberg, pp. 24-30.

Marjanovic, O. (2005), "Towards is supported coordination in emergent business processes", *Business Process Management Journal*, Vol. 11 No. 5, pp. 476-487, doi: 10.1108/14637150510619830.

Meyer, R. (2010), "Knowledge visualization", *Trends in Information Visualization*, Vol. 23, pp. 23-30.

OMG (2014), "Case management model and notation: Version 1.0: formal/2014-05-05", OMG, available at: http://www.omg.org/spec/CMMN/1.0/

Oppl, S. and Stary, C. (2019), *Designing Digital Work: Concepts and Methods for Human-Centered Digitization*, Springer Nature.

Papavassiliou, G., Mentzas, G. and Abecker, A. (2002), "Integrating knowledge modelling in business process management", In ECIS 2002 *Proceedings*, Paper 39, available at: http://aisel.aisnet.org/ecis2002/39/

Paulheim, H. (2017), "Knowledge graph refinement: a survey of approaches and evaluation methods", *Semantic Web*, Vol. 8 No. 3, pp. 489-508.

Pesic, M. and van der Aalst, W.M.P. (2006), "A declarative approach for flexible business processes management", in Hutchison, D. (Ed), *International conference on Business Process Management Workshops*, Lecture Notes in Computer Science. *Berlin and Heidelberg*, Springer Berlin Heidelberg, Vol. 4103. pp. 169-180, doi: 10.1007/11837862{\textunderscore}18.

Polleres, A. (2019), "How do linked data, open data, and knowledge graphs interplay?", in *Database and Expert Systems Applications: 30th International Conference (DEXA 2019), Proceedings, Part I*, Springer, p. xviii.

Pujara, J., Miao, H., Getoor, L. and Cohen, W. (2013), "Knowledge graph identification", In *International Semantic Web Conference*, Springer, pp. 542-557.

Reichert, M. and Weber, B. (2012), *Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, Technologies*, Springer, Heidelberg.

Tergan, S.O. (2005), "Digital concept maps for managing knowledge and information", *Knowledge and Information Visualization*, Springer, pp. 185-204.

Tergan, S.O., Keller, T. and Burkhard, R.A. (2006), "Integrating knowledge and information: digital concept maps as a bridging technology", *Information Visualization*, Vol. 5 No. 3, pp. 167-174.

Ter Hofstede, A.H., Van der Aalst, W.M., Adams, M. and Russell, N. (Eds) (2010), *Modern Business Process Automation: YAWL and Its Support Environment*, Springer, Heidelberg.

Trætteberg, H. (2008), "UI design without a task modeling language–using BPMN and diamodl for task modeling and dialog design", *Engineering Interactive Systems*, Springer, pp. 110-117.

Treisman, A. (1985), "Preattentive processing in vision", *Computer Vision, Graphics, and Image Processing*, Vol. 31 No. 2, pp. 156-177, doi: 10.1016/S0734-189X(85)80004-9.

Van der Aalst, W.M., Weske, M. and Grünbauer, D. (2005), "Case handling: a new paradigm for business process support", *Data and Knowledge Engineering*, Vol. 53 No. 2, pp. 129-162.

Villazon-Terrazas, B., Garcia-Santa, N., Ren, Y., Faraotti, A., Wu, H., Zhao, Y., Vetere, G. and Pan, J.Z., (2017), "Knowledge graph foundations", *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, Springer, pp. 17-55.

Vom Brocke, J., Hevner, A.R. and Maedche, A. (2020), "Introduction to design science research", in vom Brocke, J., Havner, A. and Maedche. A. (Eds), *Design Science Research*, Progress in IS, Springer, Cham, pp. 1-13, doi: 10.1007/978-3-030-46781-4_1.

Weske, M. (2012), *Business Process Management. Concepts, Languages, Architectures*, Springer.

Zeleny, M. (1987), "Management support systems: towards integrated knowledge management", *Human Systems Management*, Vol. 7 No. 1, pp. 59-70.

Zins, C. (2007), "Conceptual approaches for defining data, information, and knowledge", *Journal of the American Society for Information Science and Technology*, Vol. 58 No. 4, pp. 479-493.

Corresponding author
Dagmar Auer can be contacted at: dagmar.auer@jku.at