A Framework for the Intelligent Delivery and User-Adequate Visualization of Process Information*

Markus Hipp
Group Research & MBC
Development, Daimler AG,
Ulm, Germany
markus.hipp
@daimler.com

Bernd Michelberger
University of Applied Sciences
Ravensburg-Weingarten,
Weingarten, Germany
bernd.michelberger
@hs-weingarten.de

Bela Mutschler
University of Applied Sciences
Ravensburg-Weingarten,
Weingarten, Germany
bela.mutschler
@hs-weingarten.de

Manfred Reichert
University of Ulm, Institute of
Databases and Information
Systems, Ulm, Germany
manfred.reichert
@uni-ulm.de

ABSTRACT

A continuously increasing amount of data makes it difficult for knowledge-workers to identify the information they need to perform their tasks in the best possible way. Particularly challenging in this context is the alignment of process-related information (e.g., working instructions, best practices) with business processes. In fact, process-related information (*process information* for short) and business processes are usually handled separately. On one hand, shared drives, databases, and information systems are used to manage process information, on the other, process management technology provides the basis for managing business processes. In practice, enterprises often establish (Intranet) portals to connect both perspectives. However, such portals are not sufficient. Reasons are that process information is usually delivered without considering the current work context and business processes are presented to process participants in a rather static manner. Therefore, enterprises crave for new ways of making process information available. This paper picks up this challenge and presents the *niPRO framework*. niPRO is based on semantic technology and enables the intelligent delivery and user-adequate visualization of comprehensive process information.

Categories and Subject Descriptors

H.4.1 [**Information Systems Applications**]: Office Automation—*Workflow Management*

Keywords

process-oriented information logistics, process visualization, process navigation

1. INTRODUCTION

Today's knowledge-workers are confronted with a continuously increasing amount of data [3]. Examples of such data include e-mails, office files, process descriptions, web data, forms, checklists, working guidelines, and best practices. These artifacts are provided using shared drives, databases, enterprise applications, or Intranet portals.

However, employees do not only need access to data, but require context-aware information, i.e., organized data used for a specific purpose and in a specific context [21]. Thereby, identifying needed information is more time-consuming and complex than just managing data [25]. For example, often encountered problems are incomplete, incorrect, irrelevant, unpunctual, or outdated information [17]. Another major challenge is to identify the process information required to perform business processes in the best possible way.

In practice, business processes (e.g., automotive engineering [20] or patient treatment [14]) are characterized by hundreds or thousands of process steps, numerous process variants, and large amounts of related process information. As response, enterprises often establish Intranet portals as a central point of access to process information and business processes.

Existing studies [26] show that such conventional portals often contain complex and static content, rather disturbing than supporting process participants. In fact, users have different perspectives on process information and business processes. In particular, different work contexts of process participants must be considered. A less experienced employee, for example, needs more detailed working instructions than an experienced one.

This paper picks up this challenge and presents the *niPRO framework*, which enables the intelligent delivery and user-adequate visualization of process information. Specifically, the framework enables the process-oriented and context-aware delivery of process information to knowledge-workers

---

*This paper was done in the niPRO research project. The project is funded by the German Federal Ministry of Education and Research (BMBF) under grant number 17102X10.

SAC'13 March 18-22, 2013, Coimbra, Portugal.

and decision-makers, the user-adequate visualization of process information, and the intuitive and effective navigation in complex business processes. The niPRO framework is particularly suitable for knowledge-intensive business processes involving large amounts of process information, user-interactions, expertise, and decision-making [4].

The remainder of this paper is organized as follows. Section 2 provides background information. Section 3 introduces the niPRO framework in detail. Section 4 presents the application and validation of our framework based on two use cases. Section 5 summarizes related work and Section 6 concludes the paper with a summary and an outlook.

2. BACKGROUND INFORMATION

The presented research is performed in the *niPRO project*, in which we apply semantic technology (e.g., semantic networks, semantic search, and semantic analysis) to realize intelligent, user-adequate *process information portals*. The overall project goal is to support knowledge-workers and decision-makers with personalized process information depending on their current work context.

The niPRO framework itself is based on two pillars (cf. Fig. 1): *Process-oriented Information Logistics* (POIL) [19] and *Process Navigation and Visualization* (ProNaVis) [10].

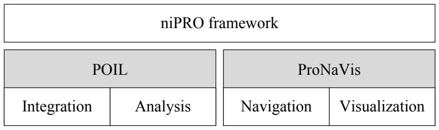

Figure 1: Pillars of the niPRO framework.

POIL targets at the provision of the right process information, in the right format and quality, at the right place, at the right point in time, and to the right actors. The latter should not actively search for needed process information anymore, but be automatically supplied with relevant process information, even if their work context is dynamically changing.

In turn, *ProNaVis* aims at enabling flexible navigation within complex business processes and related process information. ProNaVis applies innovative visualization concepts to present and deliver process information and business processes in a user-adequate manner.

Section 3 illustrates how the two approaches work together to allow for an intelligent delivery and user-adequate visualization of process information.

3. THE NIPRO FRAMEWORK

Figure 2 shows a schematic representation of the niPRO framework. It comprises four main layers: *integration*, *analysis*, *navigation*, and *visualization* (cf. Layers A-D on the left of Fig. 2). These layers and their interplay are described in detail in the following sections.

3.1 Layer A: Integration

The *integration* layer integrates data from different data sources (cf. Fig. 2a) and realizes a uniform view on these data. We distinguish between data sources of *process objects* (i.e., business processes), *information objects* (i.e., process information), and *context objects* (i.e., context information) (cf. Figs. 2b-d).

Process objects are process elements such as tasks, gateways, events, and sequence flows (according to the Business Process Model and Notation (BPMN) terminology). Note that business processes are considered at both the process schema and the process instance level. A process schema is a reusable business process template (e.g., describing patient examination processes in general) comprising, for example, tasks and sequence flows. In turn, a process instance (e.g., an examination of a certain patient) is an instance that is concurrently executed with other instances of the same or other process schemas [24]. *Information objects* refer to process information needed when working on business processes. Examples include e-mails, office files, informal process descriptions, or best practices. Finally, *context objects* represent information characterizing the work context of a process participant such as user name, roles, experiences, current tasks, used devices, locations, and time [18].

For each data source, at least one *interface* has to be implemented. Interfaces transform proprietary data objects into generic process, information, or context objects. All generic objects follow the same structure and comprise attributes such as id, url, author, file format, or raw content (e.g., the entire text of an e-mail, the coordinates of a user's position). The uniform object structure is a prerequisite to accomplish the syntactical and semantical analyses required. Specific results of the integration are three independent object spaces: the *process object space*, the *information object space*, and the *context object space* (cf. Figs. 2e-g). An object space (OS) can be defined as a set of generic process, information, and context objects (o): $OS = \{o_1, o_2, ..., o_n\}$.

3.2 Layer B: Analysis

The mentioned object spaces constitute the foundation for the *analysis* layer. The main purpose of this layer is to create a *semantic information network* (SIN) (cf. Fig. 2j) based on the information and process object space.

A SIN for the niPRO framework is constructed and maintained in six consecutive phases (cf. Fig. 2h): (1) integration of process objects, (2) integration of information objects, (3) identification of process object relationships, (4) identification of information object relationships, (5) identification of cross-object relationships, and (6) maintenance. In [19], we have described these phases in detail.

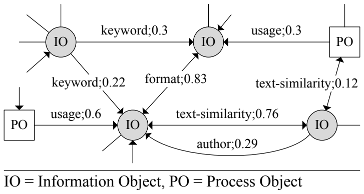

Figure 3: Simplified part of a SIN.

Figure 3 shows a simplified part of a SIN. As can be seen, the SIN does not only comprise information (i.e., gray circles) and process objects (i.e., white squares), but also relations (i.e., black arrows) between these objects. Relations may exist between information objects (e.g., a file similar to another one), between process objects (e.g., an event triggering a task), and between information and process objects (e.g., a file required for the execution of a process step). Relations are labeled with the reason of the relation and

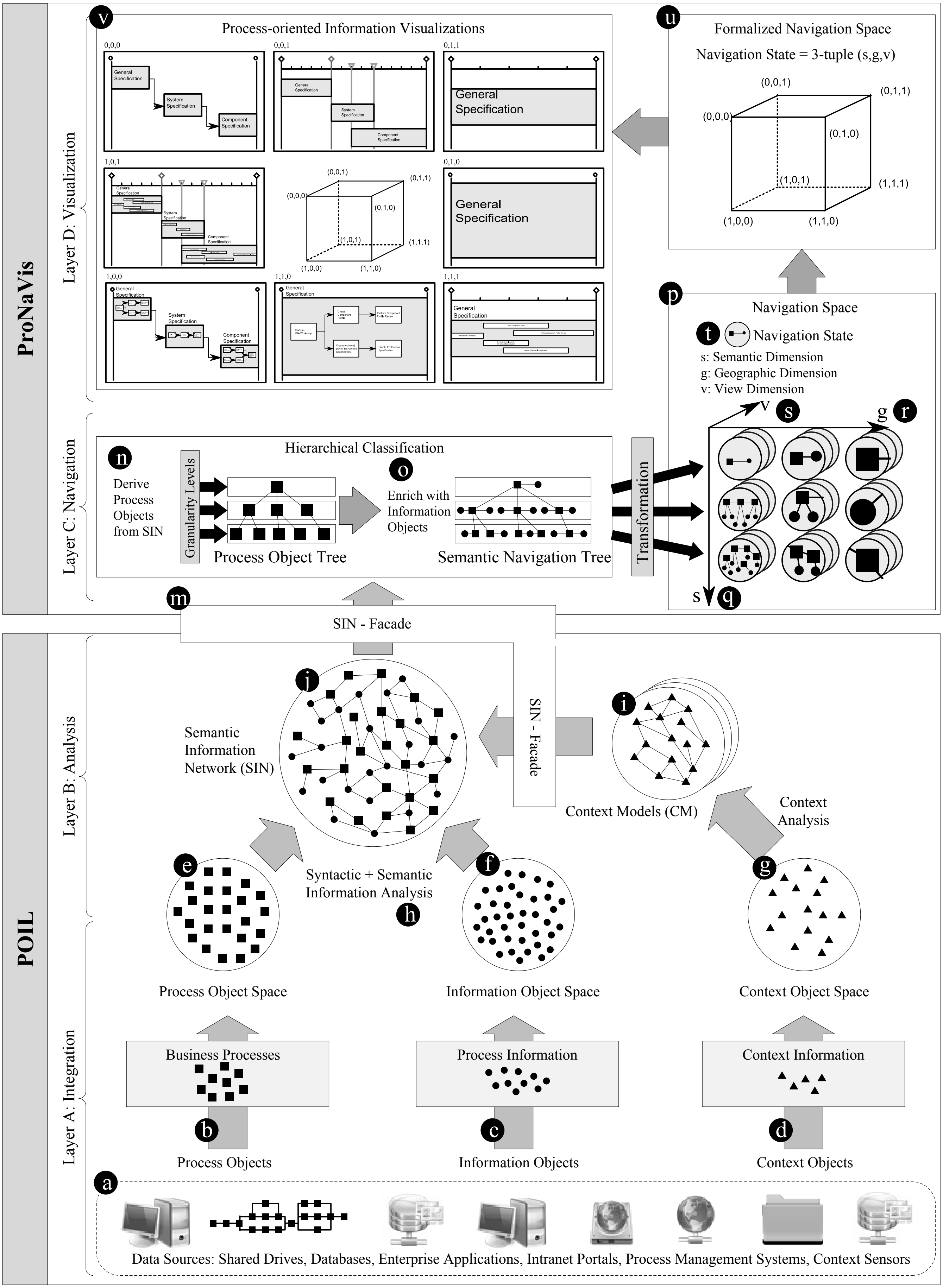

Figure 2: The Big Picture.

weighted with its relevance. This allows determining why objects are related and how strong their relation is.

For identifying the relations between objects we use a combination of syntactical and semantical analyzes[1] (cf. Fig. 2h). More precisely, algorithms from the fields of data mining, text mining (e.g., text preprocessing, linguistic preprocessing, vector space model, clustering, classification, information extraction) [11], pattern-matching, and machine learning (e.g., supervised learning, unsupervised learning, reinforcement learning, transduction) are applied. Specific algorithms are (inverse) term frequency algorithms, link popularity algorithms, and utilization context algorithms. Due to space limitations, we cannot describe them here in detail; instead of refer to [27].

In summary, a SIN can be defined as a labeled and weighted digraph $SIN = (V, E, L, W, f_l, f_w)$, where V is the set of objects (vertices), E the set of relations (edges), L the set of labels, W the set of weights, f_l the labeling function, and f_w the weighting function. The labeling function $f_l : E \rightarrow L$ assigns to each relation $e \in E(SIN)$ a label $f_l(e)$. In turn, the weighting function $f_w : E \rightarrow W$ assigns to each relation $e \in E(SIN)$ a weight $f_w(e) = [0, 1]$.

In addition to the SIN, a *context model* (CM) (cf. Fig. 2i) is constructed based on available context objects [18]. Our context model is an ontology-based model and uses pre-defined context factors such as user, location, or time [18]. The CM allows characterizing the work context of a process participant, which can then be used to filter the SIN. Based on this, the identification and delivery of currently needed process information becomes more accurate and user-oriented (as the delivery of process information can be adapted to the used device or to the experience level of a user).

The CM is completely independent from the SIN, i.e., context objects are only stored in the CM, but not in the SIN. Hence, there exists a central SIN for all users, but a specific CM for each user. Like the SIN, the CM is a labeled and weighted digraph $CM = (V, E, L, W, f_l, f_w)$, where V is the set of objects (vertices), E the set of relations (edges), L the set of labels, W the set of weights, f_l the labeling function, and f_w the weighting function. The labeling function $f_l : E \rightarrow L$ assigns to each relation $e \in E(CM)$ a label $f_l(e)$. In turn, to each relation $e \in E(CM)$, the weighting function $f_w : E \rightarrow W$ assigns a weight $f_w(e) = [0, 1]$.

The CM is applied to the SIN by the *SIN facade* (cf. Fig. 2m). The latter constitutes an interface to retrieve both process information (e.g., office files, working instructions, forms) and process objects (e.g., tasks, gateways) taking the user's working context into account. Thereby, we distinguish between an *explicit* and an *implicit* information demand. Examples of an explicit information demand include *full-text retrieval* (e.g., delivery of medical reports of a patient using the search query "John Doe report"), *concept-based retrieval* (e.g., delivery of files dealing with a certain concept like the disease "diabetes"), or *graph-based retrieval* (e.g., delivery of related process information to a certain process schema). An example of an implicit information demand is *context-based retrieval*; e.g., a patient record is delivered based on the doctor's location, i.e., the user's work context is used to retrieve information and process objects.

3.3 Layer C: Navigation

Different users in different roles need different process information from the SIN while performing their daily work. A requirements engineer, for example, needs detailed process information like checklists, guidelines, and task descriptions. In turn, a project manager asks for process information on a more abstract level, e.g., a management summary or an overview of all currently running process steps [9]. Note that even two users with the same role, but different experience might need different process information to perform the same task. We are able to gather user information from the CM. It provides user-dependent information, such as the date of joining the company, which can then be used to determine the level of experience. Our goal is to continuously deliver personalized sets of process information to employees along their dynamically changing work context. To achieve this goal, the *ProNaVis* sub-framework introduces an advanced navigation concept enabling the continuous construction and provision of these personalized sets of process information (along business processes) as well as the navigation between them (cf. Layer C in Fig. 2).

The ProNaVis navigation concept builds on an existing navigation concept - Google Earth. However, we extend the mechanisms known from Google Earth and introduce three independent navigation dimensions [8]: a *geographic*, *semantic*, and *view* navigation dimension. The *geographic dimension* allows for visual zooming without changing the level of information granularity. As example, consider Figure 4 showing a process with three sub-processes. Using the geographic navigation dimension, for instance, the user may zoom into the first sub-process *General Specification* (cf. Fig. 4a). A metaphor reflecting this dimension is the use of a magnifier. The *semantic dimension* allows displaying process information at different levels of granularity. For example, assuming that sub-processes comprise multiple process steps, the latter may be additionally displayed by adjusting the semantic dimension (cf. Fig. 4b). Finally, the *view dimension* allows users to emphasize specific process information, while reducing other. For example, the view may change from a logic-based to a time-based one (cf. Fig. 4c). Arrows between single sub-processes indicate logical relations in the logic-based view, whereas a timeline and different lengths of sub-processes are used for the time-based view.

Taken together, the described navigation dimensions constitute a *navigation space* (cf. Fig. 2p). Within this navigation space, a *navigation state* corresponds to a set of personalized process information. We use linear algebra to formalize the navigation space (cf. Fig 2u) [10]. Hence, a navigation state corresponds to a point in a Cartesian coordinate system. Unit vectors represent state transitions triggered by user interactions (adjusting levels of the navigation dimensions). We denote a set of user interactions within the navigation space as *process navigation* (PN). PN can be defined as 4-tuple $PN = (BM, NM, NS_0, NavSeq)$. BM constitutes the *basis model*, i.e., all potential navigation states within the navigation space. State transitions (or user interactions) are defined by the navigation model NM. A start navigation state is defined as NS_0, and the path the user takes through the navigation space is defined as navigation sequence $NavSeq$ (see [10] for details).

The navigation space is derived from the SIN along three steps. Steps 1 and 2 mainly deal with the derivation of the

---

[1] These analyzes are provided by and realized with a semantic middleware [28].

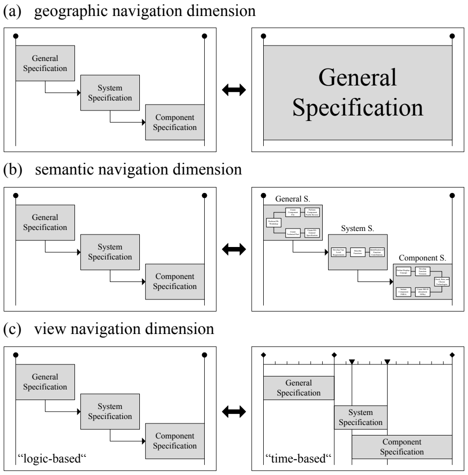

Figure 4: Different navigation dimensions.

semantic dimension, which constitutes the biggest challenge. The geographic and view dimension are derived in Step 3.

Step 1: Constructing a Process Object Tree.

In the first step, a *process object tree* is constructed (cf. Fig. 2n). This tree structure is a prerequisite to derive the semantic dimension in subsequent steps. The process object tree is built using the process objects from the SIN. Each tree level unifies all process objects with same type (e.g., pools, lanes, tasks etc.) and represents one level of information granularity. In other words, the types of process objects are used to identify and construct the levels of the process object tree. Figure 5 shows an example. The underlying process schema comprises one pool (P1), two lanes (L1, L2), three tasks (T1, T2, T3), and two data objects (D1, D2). Process information is involved by means of three files (PI1, PI2, and PI3; e.g., an e-mail, a guideline, and a template). They are integrated into the SIN as information objects (IO1, IO2, and IO3). Together, these objects form the SIN. Based on this SIN, we can now construct the process object tree. As the entity relation between pools and lanes is $1 : n$ in our example, pools are identified as a more abstract type and are thus classified on a more abstract level of information granularity (cf. Step 1 in Fig. 5). Note that we also consider other algorithms for classifying process types, which cannot be presented here due to space limitations.

Step 2: Constructing a Semantic Navigation Tree.

In a second step, the process object tree is enriched with information objects from the SIN (cf. Fig. 2o). This becomes necessary since information objects must be provided to the user. The extended process object tree is called *semantic navigation tree.*

Specifically, information objects are assigned to the levels of the process object tree according to their semantic relations to process objects (which are documented in the SIN). Consider Step 2 in Figure 5. Information object IO2, for example, is related to process object P1 in the SIN; i.e., IO2 provides information regarding P1 and must thus be classified into the same level of information granularity as P1 (level 1). Another example is IO1, representing a role description document, which provides information regarding a specific lane (in this case L2). According to this relation, IO1 is assigned to level 2 as well.

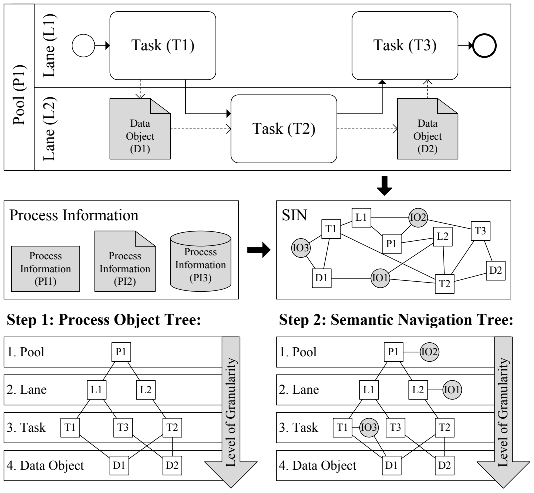

Figure 5: Deriving the semantic navigation tree.

Single information and process objects within the semantic navigation tree are represented by a uniform object structure (cf. Section 3.1). We denote this structure as *object container* (*oc*). An object container embraces all information (or properties) about the represented information or process object. Further, an object container is defined as 3-tuple $oc = \{A_{fi}, A_{fl}, CI\}$. A_{fi} is a set of mandatory attributes, such as id, title, or status. Mandatory objects are equal for information and process objects. A_{fl} is a set of optional attributes enabling type-specific object processing. As examples consider a due date assigned to a process step (i.e., a process object) or the format of a document (i.e., an information object). Finally, CI is plain text describing the actual content of a process or information object. Think of the name of a process step or the textual content of an office document.

Step 3: Constructing the Navigation Space.

All object containers on the same level of the semantic navigation tree together represent a *navigation state* (cf. Fig. 2t) in the navigation space. The semantic dimension of the navigation space (cf. Fig. 2q) can be defined using these navigation states. More precisely, each level of information granularity (i.e., each level of the semantic navigation tree) corresponds to a level in the semantic navigation dimension.

Picking up the example from Steps 1 and 2, Figure 6 illustrates Step 3: The first navigation state consists of P1 and IO2, and the second one of all information containers from the second level of information granularity (L1, L2, IO1). The arrows on the right hand side indicate that certain object containers can be inherited to other navigation states. This becomes important for the structure of the visualization (cf. Layer D in Fig. 2). For example, when a user wants to see his current task (e.g., T2 on level 3), visualizing the lane the process step belongs to (L2) would give the user a better overview.

In turn, the geographic dimension (cf. Fig. 2r) is defined by a parameter (a natural number), which is interpreted and used by the visualization layer (cf. Layer D in Fig. 2).

It indicates the zoom level to be applied to the navigation states when visualizing them.

Finally, the view dimension is constructed by combining navigation states with specific visualization rule-sets. The latter may include information about how to visualize certain attributes of different object containers (e.g., different forms, colors or fonts). For example, start and due date of a process step may be displayed as plain text in one level of the view dimension, but be implicitly displayed by presenting process boxes of different length, representing this information, in another level. Generally, it is important to provide flexible views, since different companies in different domains may have different corporate requirements concerning the visualization of their business processes.

Having added the view dimension (cf. Fig. 2s), the 3-dimensional *navigation space* (cf. Fig. 2p) is completed.

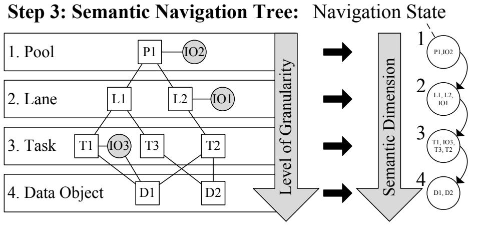

Figure 6: Deriving the semantic dimension.

3.4 Layer D: Visualization

Each navigation state as well as its associated object containers must be adequately visualized for users. Generally, the ProNaVis framework supports arbitrary visualization concepts. Figure 2v shows an example of visualizing a simple navigation space with eight navigation states. Two levels in the semantic, two levels in the geographic, and two levels in the view dimension (logic-based view: 0, time-based view: 1) are considered. Navigation state (0,0,0) shows three subprocesses on an abstract geographic and semantic level (with both levels being 0). The view is logic-based, and logic predecessor/successor relations are presented as arrows. Moving to navigation state (0,0,1) results in a time-based view, i.e., a timeline is now shown and the length of the process boxes corresponds to their duration. In order to get more detailed information, the user may navigate to navigation state (1,0,1), in which single process steps within each sub-process are shown. Finally, by adjusting the geographic dimension, the user may zoom into one sub-process to visualize corresponding process steps (this corresponds to navigation state (1,1,1)). For example, a requirements engineer benefits from this detailed navigation state, since process information is provided in the granularity level needed. In turn, a manager is free to navigate to any other (e.g., more abstract) navigation state within the SIN.

4. USE CASES

This section illustrates the application of the niPRO framework in two domains. We present two use cases, one from the clinical and one from the automotive domain.

Use Case 1: Healthcare Scenario.

The first use case stems from the clinical domain. It is based on results of a case study we performed at a large German university hospital [9, 16]. It deals with the prescription, procurement, and administration of drugs. The underlying process is knowledge-intensive, i.e., it comprises steps such as examinations and diagnosis, and involves a multitude of process information (e.g., patient records, laboratory reports, medical orders), users-interactions (e.g., patient examination, drug administration), expertise, and decision-making (e.g., drugs to be prescribed for a patient).

Specifically, the use case deals with the patient examination for which the doctor needs access to patient records, medical notes, and laboratory reports. In current practice, patient records include extensive and various kinds of data (e.g., master and transaction data, department-specific data, historical data). However, when examining a patient, usually, only small parts of a record (e.g., former diseases, pre-existing conditions, course of disease) are actually needed.

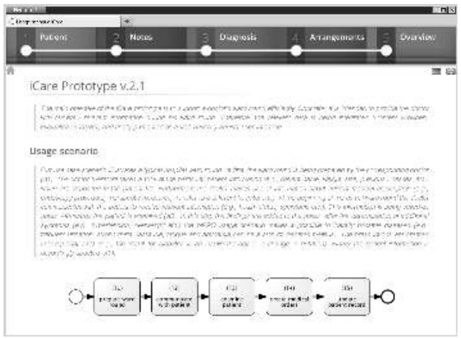

Figure 7: Screenshot iCare.

For this use case, we developed a proof-of-concept web application (iCare[2]) based on Java and a commercial semantic middleware [28] (cf. Fig. 7).

The niPRO framework allows providing exactly the process information needed. First, according to Section 3.1, process information (e.g., patient record, medical notes), business processes (e.g., examination of a patient), and context information (e.g., role, user name, used device) are integrated (cf. Layer A in Fig. 2). Following this, the SIN and CM are constructed (cf. Layer B in Fig. 2). Then, the SIN is filtered by the CM, i.e., process information is selected according to the doctor's work context (context-based retrieval). Note that the process information to be provided in this context corresponds to a specific navigation state within the navigation space. The doctor may then navigate according to the three navigation dimensions to customize visualization of process information (cf. Layer C in Fig. 2). For example, by changing the view dimension the doctor may choose a time-based view, for which he can graphically see the drug-intake of Mr. Doe within a certain period of time using a Gantt diagram (cf. Layer D in Fig. 2). Another view is the therapeutic view, in which textual suggestions on possible therapies are made.

By decreasing the level in the geographic dimension (i.e., by zooming out of the current process step), other patient examinations with the same diagnosis (i.e., other process instances) appear and thus facilitate the comparison of important information from different patients. Increasing the level of the geographic level again (i.e., by zooming in), would lead

---

[2] A screencast explaining the web application can be found on http://nipro.hs-weingarten.de/screencast.

to the concrete information about Mr. Doe. The semantic dimension could help to make the doctor's ward round more effective. In turn, decreasing the semantic level leads to a simple overview including only the most important information, e.g., the disease, current medication, blood pressure, and latest laboratory findings of Mr. Doe. This avoids information overload known from paper-based patient records. In turn, increasing the semantic level results in more detailed information, e.g., inclusion of former medications. Since this detailed information is needed rather infrequently, an overview (i.e., a more abstract level in the semantic dimension) is preferred in most cases.

In summary, the niPRO framework allows providing relevant patient records according to the doctor's work context. For example, if patient John Doe is in Room 301 and the doctor enters the room, he automatically gets the patient record of John Doe (e.g., on his tablet). In turn, the granularity level of the patient record depends on the user's role and the current process step, i.e., only those parts of the patient record are provided that are necessary for the doctor when examining the patient.

Use Case 2: Automotive Scenario.

Our second use case stems from the automotive domain. It deals with the writing of a component specification for an anti-lock braking system (ABS)-control unit. The underlying business process, called *requirements engineering* (RE), comprises three sub-processes: *general specification*, *system specification*, and *component specification*. These sub-processes are knowledge-intensive as process information is widely spread across different data sources (e.g., engineering databases, shared drives etc.) meaning that process information must be collected before the requirement engineer may actually write the component specification.

In particular, this use case deals with one part of the component specification: the access of the control unit to the controller area network (CAN) bus, a network that different control units use to communicate[3]. To avoid data loss on the CAN bus, access to the network must be coordinated among the components involved. For this purpose, component specifications of older ABS control units as well as other components must be taken into account. Thereby, the *feature list*, i.e., the list of features to be implemented, and the protocols of RE workshops adopt a key role. A prototype supporting this specific use case is currently being developed.

So far, most of the process information needed has had to be manually identified. The niPRO framework supports an automatic delivery of process information in this case. Process information from different data sources is integrated into the SIN, e.g., from projects related to the ABS control unit, but also from other components, such as engine electronics management (cf. Layer A in Fig. 2). Provided process information may also include specification documents from different components, and the parts dealing with topic "CAN-bus". Moreover, a CM is created, taking the role (e.g., requirement engineer), project description (e.g., ABS control unit), and user name to query the SIN.

The process step *write component specification*, and related process information form a navigation state. To see all related process information, the requirement engineer decreases the level of the geographic dimension (i.e., he zooms out of the current process step). Now the surrounding (e.g., the predecessing and successing) process steps *perform RE-workshop* and *integrate component specification into system specification* are presented. The requirements engineer may also switch to a logic-based view, in which successor/ predecessor relations between process steps are emphasized (cf. Layer C in Fig. 2). In this case, the feature list, workshop protocols, and personal notes are related to process step *perform requirements workshop*. Related to the current process step *write component specification* are the specification template as well as component specifications of older ABS control units and other components.

5. RELATED WORK

In an empirical case study, Bucher and Dinter [2] assess benefits, design factors, and realization approaches for conventional information logistics (IL). Heuwinkel and Deiters [7] demonstrate the possibilities and advantages of IL in the healthcare sector. Context-awareness in IL is discussed, for example, by Lundqvist et. al [15], whereas Haftor et. al [6] conducted a comprehensive literature study in the field of information logistics identifying fundamental research directions. Two of these research directions are of particular relevance in the niPRO context: *user-driven information supply* [7] and *supply of analytical information* [2].

Navigation concepts for complex information spaces exist for *zoomable user interfaces* [23]. They allow for a decreasing fraction of an information space with an increasing magnification. A framework applying these concepts is *JAZZ* [1]. Furthermore, respective user interface concepts include *Squidy*, a zoomable design environment for natural user interfaces [13], and *ZEUS*, a zoomable explorative user interface for searching and presenting objects [5]. We use these concepts and apply them to business processes.

In the area of business process visualization, the *Proviado* framework applies aggregation and reduction techniques to create flexible views on complex business process models [22]. Kolb et al. present different visualizations of business processes, by applying Concurrent Task Tree (CTT), which constitutes a task modeling language widely applied in the field of end-user development [12].

6. SUMMARY AND OUTLOOK

The alignment of process information with business processes is challenging for enterprises, since these two perspectives are usually handled separately. While process information is stored and managed using databases, information systems, and shared drives, process management technology is used to coordinate business processes. In this paper, we presented the niPRO framework, an advanced approach taking process information and business processes from integration to visualization into account. By using semantic technology, we enable the seamless and automated analysis and alignment of process information with business processes. We use context information to take the user's work context into account. By providing navigation concepts based on different navigation dimensions, users can navigate within business processes and related process information on different levels of granularity.

Future work includes the completion of an additional pro-

---

[3] The CAN bus system is used for signal transmission not only by the ABS control unit, but also by other components, for example, by the engine electronics management.

totype for the automotive domain. Based on it, we will be able to further validate the applicability of our framework and to perform advanced user experiments.

7. REFERENCES

[1] B. B. Bederson, J. Meyer, and L. Good. *Jazz: An Extensible Zoomable User Interface Graphics Toolkit in Java.* in: Proc. 13th ACM Symp. on User Interface Software and Technology, pp. 171-180, 2000.

[2] T. Bucher and B. Dinter. *Process Orientation of Information Logistics - An Empirical Analysis to Assess Benefits, Design Factors, and Realization Approaches.* in: Proc. 41st Annual Hawaii Int'l Conf. on System Sciences, pp. 392-402, 2008.

[3] A. Edmunds and A. Morris. *The Problem of Information Overload in Business Organisations: A Review of the Literature.* in: Int'l J. of Information Management, 20(1), pp. 17-28, 2000.

[4] N. Gronau, C. Müller, and R. Korf. *KMDL - Capturing, Analysing and Improving Knowledge-Intensive Business Processes.* in: J. of Universal Computer Science, 11(4), pp. 452-472, 2005.

[5] F. Gundelsweiler, T. Memmel, and H. Reiterer. *ZEUS - Zoomable Explorative User Interface for Searching and Object Presentation.* Symposium on Human Interface (HCI), LNCS 4557, pp. 288-297, 2007.

[6] D. M. Haftor, M. Kajtazi, and A. Mirijamdotter. *A Review of Information Logistics Research Publications.* in: Proc. 3rd Workshop on Information Logistics and Knowledge Supply (ILOG 2010), pp. 244-255, 2010.

[7] K. Heuwinkel and W. Deiters. *Information Logistics, E-Healthcare and Trust.* in: Proc. Int'l Conf. e-Society (IADIS'03), pp. 791-794, 2003.

[8] M. Hipp, B. Mutschler, and M. Reichert. *Navigating in Process Model Collections: A new Approach Inspired by Google Earth.* in: Proc. 1st Int'l Workshop on Process Model Collections (PMC'11), LNBIP 100, pp. 87-98, 2011.

[9] M. Hipp, B. Mutschler, and M. Reichert. *On the Context-aware, Personalized Delivery of Process Information: Viewpoints, Problems, and Requirements.* in: Proc. 6th Int'l Conf. on Availability, Reliability and Security (ARES'11), pp. 390-397, 2011.

[10] M. Hipp, B. Mutschler, and M. Reichert. *Navigating in Complex Business Processes.* in: Proc. 23rd Int'l Conf. on Database and Expert Systems Applications (DEXA'12), LNCS 7447, pp. 466-480, 2012.

[11] A. Hotho, A. Nürnberger, and G. Paaß. *A Brief Survey of Text Mining.* in: J. for Computational Linguistics and Language Technology, 20(1), pp. 19-62, 2005.

[12] J. Kolb, M. Reichert, and B. Weber. *Using Concurrent Task Trees for Stakeholder-centered Modeling and Visualization of Business Processes.* S-BPM ONE 2012, CCIS 284, pp. 237-251, 2012.

[13] W. A. König, R. Rädle, and H. Reiterer. Squidy: a zoomable design environment for natural user interfaces. in: Proc. 27th Int'l Conf. on Human Factors in Computing Systems (CHI'09), pp. 4561-4566, 2009.

[14] R. Lenz and M. Reichert. *IT Support for Healthcare Processes - Premises, Challenges, Perspectives.* in: Data & Knowledge Eng., 61(1), pp. 39-58, 2007.

[15] M. Lundqvist, K. Sandkuhl, T. Levashova, and A. Smirnov. *Context-Driven Information Demand Analysis in Information Logistics and Decision Support Practices.* in: Proc. 1st Int'l Workshop on Contexts and Ontologies: Theory, Practice and Applications, pp. 124-127, 2005.

[16] B. Michelberger, B. Mutschler, and M. Reichert. *On Handling Process Information: Results from Case Studies and a Survey.* in: Proc. 2nd Int'l Workshop on Empirical Research in Business Process Management (ER-BPM'11), LNBIP 99, pp. 333-344, 2011.

[17] B. Michelberger, B. Mutschler, and M. Reichert. *Towards Process-oriented Information Logistics: Why Quality Dimensions of Process Information Matter.* in: Proc. 4th Int'l Workshop on Enterprise Modelling and Information Systems Architectures (EMISA'11), LNI 190, pp. 107-120, 2011.

[18] B. Michelberger, B. Mutschler, and M. Reichert. *A Context Framework for Process-oriented Information Logistics.* in: Proc. 15th Int'l Conf. on Business Information Systems (BIS'12), LNBIP 117, pp. 260-271, 2012.

[19] B. Michelberger, B. Mutschler, and M. Reichert. *Process-oriented Information Logistics: Aligning Enterprise Information with Business Processes.* in: 16th IEEE Int'l EDOC Conf. (EDOC 2012), pp. 21-30, 2012.

[20] D. Müller, J. Herbst, M. Hammori, and M. Reichert. *IT Support for Release Management Processes in the Automotive Industry.* in: Proc. 4th Int'l Conf. on Business Process Management (BPM'06), pp. 368-377, 2006.

[21] N. Mundbrod, J. Kolb, and M. Reichert. *Towards a System Support of Collaborative Knowledge Work.* in: Proc. of the 1st Int'l Workshop on Adaptive Case Management (ACM'12), 2012.

[22] M. Reichert, J. Kolb, R. Bobrik, and T. Bauer. *Enabling Personalized Visualization of Large Business Processes through Parameterizable Views.* in: 27th ACM Symp. On Applied Computing (SAC'12), pp. 1653-1660, 2012.

[23] H. Reiterer and T. Büring. *Zooming Techniques.* Ency. of Database Systems, pp. 3684-3689, 2009.

[24] S. Rinderle, M. Reichert, and P. Dadam. *On Dealing with Structural Conflicts between Process Type and Instance Changes.* in: Proc. 2nd. Int'l Conf. Business Process Management (BPM'04), pp. 274-289, 2004.

[25] J. Rowley. *The wisdom hierarchy: representations of the DIKW hierarchy.* in: J. of Information Science, 33(2), pp. 163-180, 2006.

[26] J. J. V. Wijk and W. A. A. Nuij. *Smooth and Efficient Zooming and Panning.* in: Proc. of the 9th annual IEEE conference on Information visualization (INFOVIS'03), pp. 15-22, 2003.

[27] J. Wurzer. *New approach for semantic web by automatic semantics.* European Semantic Technology Conf. (ESTC), 2008.

[28] J. Wurzer and B. Mutschler. *Bringing innovative Semantic Technology to Practice: The iQser Approach and its Use Cases.* in: Proc. 4th Int'l Workshop on Applications of Semantic Technologies (AST'09), LNI 154, pp. 3026-3040, 2009.