Noname manuscript No.
(will be inserted by the editor)

Knowledge-intensive Processes

Characteristics, Requirements and Analysis of Contemporary Approaches

Claudio Di Ciccio · Andrea Marrella · Alessandro Russo

Received: date / Accepted: date

Abstract Engineering of knowledge-intensive processes (KiPs) is far from being mastered, since they are genuinely knowledge- and data-centric, and require substantial flexibility, at both design- and run-time. In this work, starting from a scientific literature analysis in the area of KiPs and from three real-world domains and application scenarios, we provide a precise characterization of KiPs. Furthermore, we devise some general requirements related to KiPs management and execution. Such requirements contribute to the definition of an evaluation framework to assess current system support for KiPs. To this end, we present a critical analysis on a number of existing process-oriented approaches by discussing their efficacy against the requirements.

Keywords Knowledge-intensive Processes · Process Management Systems · Case Management · Process Flexibility · Process Mining

1 Introduction

Business Process Management (BPM) [25, 76] is an active area of research, which is highly relevant from a practical point of view while offering many technical challenges. It is based on the observation that each product or service that a company provides to the market is the outcome of a number of activities performed. *Business processes* are the key instruments for organizing these activities and for improving the understanding of their interrelationships [76]. BPM aims at providing techniques and softwares to design, enact, control, and analyze business processes involving humans, organizations, documents and other sources of information.

A *Process Management System* (PMS) is a software system that is driven by explicit process representations (called *process models*) to coordinate the enactment of business processes. Process models are the main artifacts for supporting process enactment through a PMS, as they provide an explicit representation of process knowledge. Consolidated approaches develop modeling activities along three main dimensions [42]: *(i)* the *control-flow* perspective, describing the structure of a process in terms of *tasks* (atomic work units that describe an activity to be performed) and the relationships between them (usually described by routing constructs like sequences, parallel and alternative branches); *(ii)* the *data perspective*, describing data elements consumed, produced and exchanged during process execution; and *(iii)* the *resource* perspective, describing the operational and organizational context for process execution in terms of resources (i.e., people, systems and services able to execute tasks) as far as their capabilities (i.e., any qualification or skill that is relevant for task assignment and execution). In addition to these dimensions, which can be considered as orthogonal to each other, there is the cross-dimensional *exception handling* perspective, that defines the approaches dealing with undesirable events that may arise. Exceptions can occur in each of the first three dimensions (e.g., incorrect process structure, task failures, missing or incorrect data, or resource unavailabilities) and handling strategies may require to act on the control-flow, data and/or resource models.

A PMS that takes in input a process model is able to manage the process routing by deciding which tasks are

C. Di Ciccio
Wirtschaftsuniversität Wien
E-mail: claudio.di.ciccio@wu.ac.at

A. Marrella · A. Russo
Sapienza Università di Roma
E-mail: {marrella,arusso}@dis.uniroma1.it

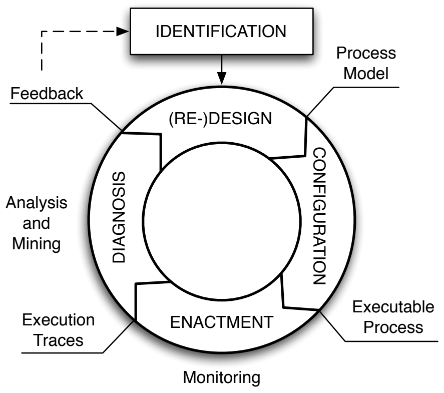

Fig. 1 The life cycle of a business process

enabled for execution and by assigning them to proper resources. A single execution of a process model within the engine of the PMS is called *process instance* [33].

PMSs hold the promise of facilitating the everyday operation of many enterprises and work environments, by supporting business processes in all the steps of their life cycle [24]. As shown in Fig. 1, the life cycle of a business process is organized in 4 main stages. In the *design* phase, starting from a requirements analysis, process models are designed using a suitable modeling language. In the *configuration* phase, process models are implemented by configuring a PMS that supports process enactment. In the *enactment* phase, process instances are then initiated, executed and monitored by the run-time environment, and performed tasks generating execution traces are tracked and logged. Finally, in the *diagnosis* phase, process logs are evaluated and mined to identify problems and possible improvements, potentially resulting in process re-design and evolution.

In previous years, well-established engineered approaches and tools have been developed for business processes, and today BPM environments provide wide support for different modeling styles and for all phases of the process life cycle [75]. Process management approaches are often based on the assumption that processes are characterized by repeated tasks, which are performed on the basis of a process model prescribing the execution flow in its entireness [49]. This kind of structured work includes mainly production and administrative processes [44]. However, the current maturity of process management methodologies has led to the application of process-oriented approaches in new challenging knowledge-intensive scenarios, such as healthcare, emergency management, projects coordination, case management, etc. In these working environments, most business functions involve collaborative features and unstructured processes that do not have the same level of predictability as the routine structured work [6]. This has led to the definition of the class of *knowledge-intensive processes* (KiPs). In KiPs, it is needed to understand the knowledge dimension of processes and to consider the role of human-centered knowledge, so as to go beyond process automation [49]. KiPs are often slightly structured and can be partially mapped to process models. Variations from structured reference models are common due to autonomous user decisions and unpredictable events. Such variations make the structure of the process less rigid, as they involve undesigned and unscheduled knowledge production.

1.1 Motivations and Research Objectives

In recent years, the need to deal with KiPs has emerged as a leading research topic in the BPM domain [21, 62], due to the prominent role that knowledge workers play in modern organizations. This is backed by both quantitative considerations, as it has been estimated that today knowledge workers represent between 25% and 40% of the workforce [9], and qualitative observations, as knowledge workers have a major impact on organizational success and value creation. Several entities, ranging from public administrations to private companies, recognize that their core processes increasingly rely on best practices rather that on explicit procedure-oriented processes. When knowledge creation, management and sharing are explicitly related to business processes, the collaborative nature of KiPs has to be considered as an integral part of practice-oriented processes [50]. BPM researchers have recently recognized the need to extend existing approaches to support KiPs and meet their challenging requirements, which actual BPM frameworks are not able to handle adequately. Specifically, the knowledge and collaboration dimensions need to be integrated with the traditional control flow/data dimensions and consider them as a whole [28] by possibly reshaping the process life cycle. Therefore, the ultimate goal of a BPM framework shifts from providing process automation to supporting decision making and collaboration between knowledge workers.

This motivational framework has led the research community to bring together research areas that have been addressing related problems from different perspectives [50]. On one side, the BPM community has largely focused on *coordination* support (relying on the foundational notion of process models), with minor emphasis of *collaboration* aspects. On the other side, the community targeting Computer Supported Cooperative Work (CSCW) has mainly focused on *collaboration* support, not necessarily framed in a process-oriented perspective. As knowledge work and KiPs combine coordination and collaboration with a knowledge dimension, the broad field of Knowledge Management (KM) has been considered too, as it allows to understand how knowledge is created, shared and used.

To date, however, the current research literature on KiPs has mainly focused on providing reference definitions for the concepts of "knowledge", "knowledge workers" and "knowledge-intensity" for a business process. These definitional frameworks, often coupled with concrete use cases that illustrate knowledge work, are typically the starting point for the identification of some high-level characteristics that contribute to make a process knowledge intensive. While it is increasingly recognized that there is a lack of a holistic system support for knowledge workers and the processes they undertake, nowadays the discussion about KiPs misses a clear mapping between characteristics and system requirements.

The main objective of this paper is to fill this gap. To this extent, *(i)* a precise characterization of KiPs is provided, *(ii)* a set of requirements for process-oriented systems aiming at their support is derived, and *(iii)* both consolidated and emerging process-oriented systems and approaches are reviewed, with an assessment of their level of support to the aforementioned requirements. As a major overall objective we aim at providing an evaluation framework that helps researchers and practitioners to understand KiPs, to contextualize and position their work and proposals, and to focus their efforts in the selection of existing approaches.

1.2 Methods and Results

In order to achieve the stated objectives, we adopt a research method, reflected in the structure of the paper, that combines scientific literature analysis with personal experience and background. As primary source of information, we consider relevant work produced by the BPM community in the area of KiPs. In particular (Section 2), we focus on research efforts that investigate the knowledge dimension in business processes, provide definitional frameworks for KiPs, identify their distinctive characteristics and discuss emerging research challenges. Then, we identify a reference definition of what a KiP represents in our view. In addition, the class of KiPs is positioned in the broad area of BPM relying on a well-known classification spectrum used for classifying processes on the basis of their degree of structure.

To root our analysis in a concrete setting, we introduce three representative application scenarios (Section 3), derived by the practical experience we gained while working in EU- or Italian- funded research projects that targeted the management of specific classes of KiPs. Starting from the characterizations of KiPs available in the literature and the representative use cases, we identify eight key characteristics of KiPs (Section 4). Then, we derive a set of 25 requirements related to KiPs management (Section 5). Requirements describe the features that must be provided by a system that wants to successfully satisfy the KiPs characteristics and contribute to the definition of an evaluation framework to assess current system support for KiPs. In particular, the requirements framework is used to evaluate a selected subset of process-oriented systems and approaches (Section 6). The analysis considers systems and approaches that: *(i)* are the expression of consolidated research activities; *(ii)* rely on formal foundations and well-established methodologies; *(iii)* had or are having a significant impact and relevance for both researchers and practitioners. Additional inclusion criteria are given by the actual availability of system implementations and/or an exhaustive reference documentation. While the authors' work does not aim at providing a comprehensive evaluation of all existing tools and approaches, the ones considered here are representative of the different process management paradigms that have emerged over the years: from activity-centric approaches (based on either imperative or declarative models) to object-aware and data-centric methodologies. Moreover, the analysis includes recent research prototypes resulting from our work, which may complement or extend the current state of the art.

From the analysis of the process-oriented approaches against the identified requirements (Section 7), it is clear that KiPs reveal some challenging features (e.g., communication-orientation, low predictability, etc.) that pose serious problems for their support through the use of existing approaches. Although each approach is able to provide the right support for single requirements, there is the lack of a holistic approach which allows to tackle the set of identified requirements as a whole and to provide a targeted support for a KiP. The realization of an approach with the above characteristics can be regarded as a key success factor for the fruitful application of BPM and represents one main challenge that is currently under investigation.

2 Understanding and Defining KiPs

The increasing interest in KiPs is reflected in the different characterizations available in literature. KiPs are inherently related to the concepts of *knowledge*, *knowledge work* and *knowledge workers*. We consider here the relationships among these concepts, relying on existing works that explore the links between knowledge and process management. As there is no unique definition of KiP, we identify a reference definition that, in our view, best represents a KiP in relation to the focus of this paper. The general discussion is then explicitly framed in the scope of BPM, in order to understand the role of KiPs in the spectrum of process management.

2.1 KiPs Definitions and Characterizations

KiPs are often related to the need of considering and understanding the knowledge dimension in business processes [49]. They are positioned in the largely unexplored intersection between the BPM and the KM fields. According to this view, knowledge has to be considered as an integral part of business processes, so as to go beyond the approaches that manage processes and process-related knowledge separately. According to Davenport, knowledge is a combination of experience, context, interpretation and reflection and involves more human participation than information [15]. He recognizes the knowledge intensity by the diversity and uncertainty of process input and output [16]. This definition suggests that process-related knowledge is strongly human-centered, emphasizing the role of so called "people components" that *"create, co-create, share, transfer and apply knowledge in the context of the processes they participate in, in order to achieve organizational goals and create value"* [49]. As detailed in [29], human-centered knowledge can be further refined in explicit and tacit knowledge. While *explicit* knowledge is easy to communicate and store, as it can be formalized and systematized in a common representation format (e.g, databases, documents, etc.), *tacit* or *implicit* knowledge derives from experience, mental models and perspectives which cannot be easily formalized or shared through an externalization process [28]. This form of "personal" knowledge, proper to the so-called "knowledge worker", is explicitly mentioned in relation to KiPs in [7], where KiPs are defined as *"task sequences, which strongly rely on the employment of tacit knowledge"*, and in [29], where it is stated that *"knowledge-intensive business processes deal very much with creating and using tacit knowledge from many participants"*, in line with the view of business processes as "knowledge-in-action or actionable knowledge" [49]. Similarly, stressing the need to leverage human expertise and knowledge in process management, in [46] Malhotra relates knowledge management to *"organizational processes that seek a synergistic combination of data and information-processing capacity of information technologies and the creative and innovative capacity of human beings"*.

The research literature has also defined several factors that are fundamental to model and execute KiPs. In [27], the authors focus on the impact that the know-how of single process participants may provide to KiPs. Different participants usually have different skills from different domains at different levels, and the resulting processes may include many innovative and creative parts difficult to be straightjacket into classical control-flows. In this direction, according to [28], a process is defined as knowledge intensive if *"its value can only be created through the fulfillment of the knowledge requirements of the process participants"*. In their exploratory study [36], the authors define KiPs as *"processes that require very specific process knowledge, typically expert involvement, that are hard to predict and vary in almost every instance of the process"*. The authors identify the main dimensions that emerge from the literature for characterizing KiPs, including the level of repeatability, predictability and complexity, the required creativity, expertise, level of decision and role of process participants as knowledge workers, the suitability for automation, and the degree of structuredness. These dimensions often contribute to the definition of classification frameworks (e.g., cf. [49]) to differentiate KiPs from other kinds of processes, similarly to the spectrum we adopt in this paper (cf. Section 2.2).

Starting from a literature analysis, a characterization of KiPs is provided in [29]. A KiP often does not cover structured working practices and includes innovative and creative parts, with a significant contribution coming from human-centered knowledge. Process-related knowledge has often a very short life-time and becomes outdated very quickly, whereas building up this knowledge is considered a time-intensive task [16]. As a consequence, the tasks, their sequencing and the event flow of KiPs are not clear from the very beginning, cannot be precisely predefined and can evolve as the process progresses and as a result of communication between process workers, considered an integral part of the process itself. In [14], the authors characterize KiPs through the definition of a Knowledge-based Business Process Ontology (KBPO). According to their view, a KiP basically includes knowledge paths and transactions. A *knowledge path* is a sequence of functions (i.e., tasks) performed by human members on a knowledge object using knowledge tools, i.e., technological artefacts that produce knowledge transformations (such as creating, combining and modifying knowledge objects). Knowledge objects, considered as any data, information and artifact that can be produced or used during process execution, are involved in *knowledge transactions*, defined as transportations of knowledge objects between two or more communicating process members. Central to the ontology is the concept of knowledge-intensive function, defined as *"one that involves decision making, requires considerable context knowledge, and its inputs and outputs are complex and dynamic"*.

Recently, a precise and holistic definition of KiPs has been provided by Vaculín et al. in [72]. In our view, their definition captures the main distinctive elements of a KiP emerged so far, combined with a process management perspective. For the scope of this paper, we thus rely on the the following definition.

Knowledge-intensive Processes (KiPs): processes whose conduct and execution are heavily dependent on knowledge workers performing various interconnected knowledge intensive decision making tasks. KiPs are genuinely knowledge, information and data centric and require substantial flexibility at design- and run-time [72].

The main scope of this class of processes is tied to the central role of knowledge workers, considered again as autonomous decision makers that collaborate with the goal of performing information and decision intensive tasks that depend on knowledge artifacts. A KiP has to provide guidance and support to users performing these tasks, in the form of contextual information, choices, recommendations and advices that facilitate decision making. The authors recognize that the overall flows of actions and knowledge is thus the result of the interplay between a *business functional perspective* and a *decision-driven process structure*. The former defines a traditional structured flow originating from procedural pattern and guidelines. The latter includes domain specific knowledge and contextual information driving user decision making and influencing the process flow.

The establishment of a definitional framework for *collaborative knowledge work* (CKW) represents the first step in the methodological approach adopted in [58]. The authors rely on well-established and consolidated definitions of *knowledge*, *knowledge work* and *knowledge workers*, which allow them to define CKW as *"knowledge work jointly performed by two or more knowledge workers in order to achieve a common business goal"*. Four key characteristics of CKW are thus identified, namely (i) uncertainty, (ii) goal orientation, (iii) emergence, and (iv) growing knowledge base. Similarly, nine dimensions to differentiate CKW scenarios are detailed, with a focus on system support. As a foundational step towards a CKW system, the authors outline the CKW life cycle as a variant of process life cycle described in Section 1. An adapted representation of the life cycle is shown in Fig. 2. Basically, the identification of the information flow, the knowledge actions and the coordination structures (*orientation* phase) is followed by the definition of a goal-oriented collaboration template (*template design* phase). The context-aware instantiation of a collaboration template (*in-*

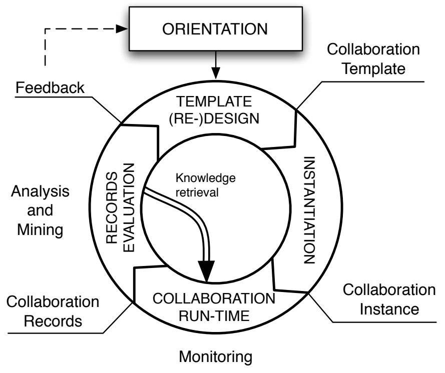

Fig. 2 Collaborative knowledge work life cycle

stantiation phase) results in a collaboration instance that supports the run-time interaction between knowledge workers (*collaboration run-time* phase). Knowledge workers may access and exploit historical collaboration records as part of the knowledge base that supports instance progression. The actual collaboration instance, in turn, produces new collaboration records that are evaluated (*records evaluation* phase) to improve the understandings gained in the initial orientation and possibly reshape the defined templates.

2.2 The Spectrum of Process Management

To better understand and position KiPs in the context of BPM, we classify business processes along a spectrum on the basis of the degree of structuring and predictability they exhibit, which directly influence the level of automation, control and support that can be provided, as well as the degree of flexibility that is required. The spectrum discussed here is inspired by and derived from process classifications presented in [31, 37, 65].

At one end of the spectrum shown in Fig. 3 there are *structured processes*, which reflect highly predictable routine work with low flexibility requirements and controlled interactions among process participants (such as production and administrative processes) [44]. Process logic is known in advance and pre-definable, in terms of the activities to be executed, their dependencies, and the resources performing the activities. As a consequence, all possible options and decisions that can be made during process enactment are captured in a process model defined a priori, which can be repeatedly instantiated in a predictable and controlled manner [55].

Structured processes with ad hoc exceptions have similar characteristics than structured processes, as they reflect operational activities that typically comply with a predefined plan. Although, the occurrence of external events and exceptions can make the structure of the process less rigid. The actual course of action

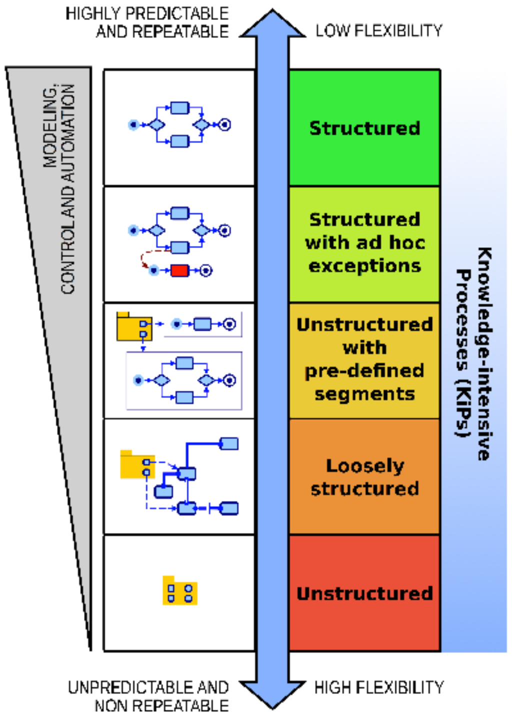
HIGHLY PREDICTABLE AND REPEATABLE
LOW FLEXIBILITY
MODELING, CONTROL AND AUTOMATION
Structured
Structured with ad hoc exceptions
Unstructured with pre-defined segments
Loosely structured
Unstructured
Knowledge-intensive Processes (KiPs)
UNPREDICTABLE AND NON REPEATABLE
HIGH FLEXIBILITY

Fig. 3 The spectrum of process management

may deviate from the predefined reference work practices and process adaptation strategies may be required [41]. In the presence of anticipated exceptions, possible deviations that can be encountered are predictable and defined in advance via exception handlers, typically prespecified into the process model (e.g., in the handling of financial back-office transactions). Conversely, unanticipated exceptions can be only detected during the execution of a process instance. Their handling typically requires ad-hoc process changes at run-time [63].

In many application domains, e.g., in the handling of insurance claims, work practices are rather unstructured and proceed on an ad-hoc basis. In *unstructured processes with predefined segments* the overall process logic is not explicitly defined, but the existence of policies and regulations allows to identify pre-definable, structured fragments. These fragments can refer to explicit, prescriptive procedures, or may take the form of underspecified templates and guidelines. Process parts that are undefined or uncertain can only be specified and incorporated in the range of the existing process model as the process evolves, and decisions regarding the specification of (parts of) the process have to be deferred. Similarly, predefined process fragments need to be selected and properly composed on a per-case basis.

A wide range of processes exhibit a *loosely structured* behavior. While work practices are not subject to prescriptive reference procedures, the existence of policies and business rules induces constraints that implicitly frame the scope of action of process participants. The set of possible activities may be known and predefined, but their execution ordering is not entirely foreseeable, as many possible execution alternatives are allowed (e.g., a patient treatment procedure depends on her/his actual physical data and the reported list of symptoms). Rather than using a procedural language for expressing the allowed sequences of activities, processes are described through the usage of constraints, that implicitly define these alternatives by prohibiting undesired execution behavior.

Finally, the spectrum ends with *unstructured processes*, characterized by a low level of predictability and high flexibility requirements. Process participants decide on the activities to be executed as well as their execution order, and the structure of a process thus dynamically evolves. These processes directly reflect knowledge work and collaboration activities driven by rules and events, for which no predefined models can be specified and little automation can be provided. Knowledge workers rely on their experience to perform ad-hoc tasks on a per-case basis and handle unexpected changes in the operational context. For processes with these characteristics, only their goal is known a priory.

The class of KiPs is transversal with respect to the classification presented here. Although the knowledge intensity generally increases along the spectrum, almost all the classes of processes discussed above may include elements that make them knowledge-intensive. The knowledge dimension may emerge, for example, in the way knowledge workers deal with unexpected exceptions. Similarly, knowledge workers put in place their experience and expertise for instantiating and concretizing underspecified procedures, or for contextually selecting and composing appropriate plan fragments. Moreover, individual and collaborative decision-making contributes to the definition of the best course of action in loosely structured or unstructured work practices.

While process structure represents the main classification dimension considered here, other process classification frameworks, summarized in [2], consider additional dimensions that are orthogonal or complementary to each other. In particular, the *degree of framing* can be correlated with the nature of process participants. Consistently with the spectrum of Fig. 3, highly repeatable processes are mainly tightly framed, i.e., they rely on a priori defined process models, with a focus on Application-to-Application (A2A) interactions. Conversely, KiPs are significantly less framed and mainly characterized as human-centric, with a predominance of Person-to-Person (P2P) interactions. Unstructured and unframed processes, with which no *explicit* process model is associated, are typically tied to the scope of groupware systems. While it is possible to identify an existing boundary between the fields of BPM and CSCW, as given by the notion of *process awareness*, we agree that pushing the boundaries of BPM to deal with KiPs contributes to reducing the gap and establish a synergy between the two fields [50]. On one side, groupware systems do not assume an explicit process perspective and mainly focus on supporting flexible collaboration. On the other side, we recognize the role that CSCW methodologies and groupware systems can play in supporting specific aspects of KiPs (in particular, the collaboration and communication needs that emerge in the orientation phase - cf. Fig. 2 - and become explicit at run-time). This view does not exclude unstructured and unframed processes from the broad class of KiPs, especially when the use of BPM techniques such as process mining can contribute to the creation of process awareness.

3 Representative Application Scenarios

Processes that are inherently knowledge-intensive can be found in several fields and domains. Research and implementation projects, criminal investigations, human resource management are all examples of domains that were subject to case studies and have been considered for the definition of scenarios and use cases for KiPs (for example, in [38, 40, 58, 72]). In this section, we present three different scenarios defined on the basis of case studies we conducted. In particular, we explore the knowledge-intensive nature of real-world processes in *(i)* the recovery and response assistance during natural or man-made disasters, *(ii)* in patient case management in a hospital, and *(iii)* in project management and scientific paper writing activities. The heterogeneity of the scenarios allows us to consider processes that cover the spectrum of process management introduced before (cf. Section 2.2), as a basis for the identification and systematization of the key characteristics of KiPs.

3.1 Emergency Management Processes

In the last years, the widespread availability of mobile computing platforms has led to the application of process-oriented approaches in *pervasive and highly dynamic scenarios.* An interesting example comes from the *emergency management* domain, where teams of first responders act in disaster locations with the main purpose of assisting potential victims and stabilizing the situation. First responders can benefit from the use of mobile devices and wireless communication technologies, as well as from the adoption of a process-oriented

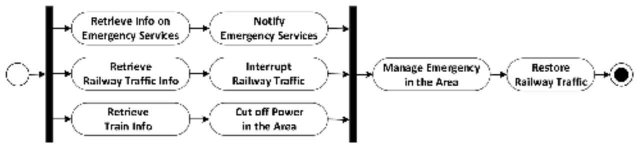
Retrieve Info on Emergency Services
Notify Emergency Services
Retrieve Railway Traffic Info
Interrupt Railway Traffic
Retrieve Train Info
Cut off Power in the Area
Manage Emergency in the Area
Restore Railway Traffic

Fig. 4 A standardize procedure for managing derailments

approach for team coordination. A response plan encoded as a business process and executed by a PMS deployed on mobile devices can help to coordinate the activities of first responders equipped with smartphones and supported by mobile networks. Starting from the experience gained in the area and lessons learned from the European project WORKPAD [11], we discuss now the main features underlying this kind of processes.

When some emergency occurs, in general there exist *standardized procedures* to be performed for dealing with the specific emergency. Such procedures often involve the execution of *basic activities* and *abstract activities*, whose exact definition may not be known until the time the procedure has started execution. For example, let us consider the emergency management procedure shown in Fig. 4 and used by the main Italian railway company to manage train derailments. The procedure starts when the railway traffic control center receives an accident notification from the train driver and begins by collecting information about the train (e.g., the area where the train derailed, the number of affected coaches, etc.) and the emergency teams available in the area. Then, it may need to cut off power in the area and interrupt railway traffic around the derailment scene. The above basic activities refer to atomic tasks whose completion allows to collect/manipulate/update information reflecting the evolution of the contextual scenario in which the procedure is under execution.

Such information (mostly unknown at design-time) may be used for defining a concrete *response plan*, which includes the set of tasks to be executed directly on the field by first responders. These tasks, abstracted into the activity “Manage Emergency in the Area”, need to be contextually and dynamically selected (or generated) at run-time, when the concrete objective of the abstract activity emerges. For example, let us suppose that the scenario in Fig. 5 reflects the contextual information collected during the execution of the basic activities. It depicts a grid-based map of the area, and it assumes that the train is composed by a locomotive (located at *loc33*) and two passenger coaches (located at *loc32* and *loc31* resp.). The team is composed of four first responders and two robots, initially located at cell *loc00*. First responders and robots provide specific skills and capabilities. For example, *act1* is able to ex-

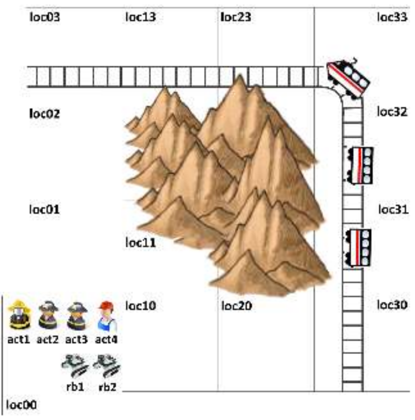

Fig. 5 Context and area of the intervention

tinguish fires, *act2* and *act3* can evacuate people from train coaches and, when a robot's battery is empty, *act4* can charge it. The two robots, instead, may take pictures and remove debris from specific locations. A concrete goal for the abstract activity "Manage Emergency in the Area" may reflect, for example, the objective of evacuating people from the coach at *loc32*, extinguishing a fire in the coach at *loc31* and taking pictures for evaluating possible damages to the locomotive.

In Fig. 6, a candidate response plan is shown, encoded as a BPMN process.[1] It matches with the context shown in Fig. 5. The process instructs *act1* to reach *loc31* in order to extinguish fire. In parallel, after the battery of robot *rb1* has been recharged by *act4*, it can move in *loc32* for removing debris while *act2* can start to evacuate people in that location. Finally, *rb1* can move into *loc33* for taking pictures. A correct execution (i.e., without exceptions) of the above process guarantees to satisfy the concrete goal condition associated to the "Manage Emergency in the Area" activity.

The design of a response plan is usually a time-consuming and error-prone activity for a process designer, since it depends on the current contextual information (the positions of first responders, the battery level of robots, etc.), and the correctness of the plan execution is highly constrained by the values (or combination of values) of each contextual data.

Furthermore, during a response plan enactment, dynamic context changes reflecting new goals to be achieved (e.g., to extinguish a fire burnt up in a coach), external events coming from the environment (e.g., the discovery of some wounded in a coach) or tasks not executed as expected (e.g., the failure of a coach evacuation) may occur continuously and invalidate the response plan under execution, by preventing the achievement of its objectives. Therefore, the adaptation of a re-

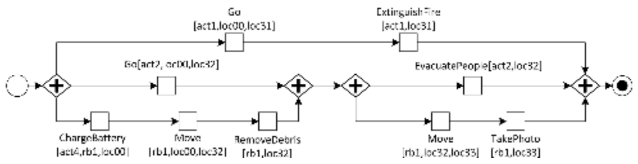

Fig. 6 A response plan dealing with the scenario in Fig. 5

sponse plan is crucial to deal with any emergent contextual change. It requires an extensive manual effort for the process designer, which has to anticipate all potential problems and ways to overcome them in advance. A relevant challenge investigated by the research literature in process flexibility [53] is how to make response plans' adaptation as automated as possible, with minimum manual intervention at run-time.

With respect to the spectrum shown in Section 2.2, a standardized procedure for dealing with a particular emergency can be seen as a pre-defined *fragment of a larger unstructured process*, which involves one or more response plans. A response plan can be seen as the best-practice process drawn up with any contextual information available at the time and may potentially range from the *structured with ad-hoc exceptions* to the *unstructured* categories, depending on the complexity and on the gravity of the emergency to deal with.

3.2 Diagnosis and Treatment Processes

In healthcare organizations, a wide range of processes with different characteristics and requirements coexist, interlinked and interleaved [43, 61, 69]. We focus here on the diagnostic and therapeutic steps driven by clinical decision making and medical case data, as representative examples of KiPs. Patient case management is highly knowledge-driven, as it depends on medical knowledge and evidence, on case- and patient-specific information, and on clinicians' expertise and experience [48]. Moreover, the delivery of complex care may involve several departments and require an active coordination and collaboration of different professionals with heterogeneous skills and expertise.

To frame the scope of our discussion, we refer here to a case study we analyzed in the context of an ongoing collaboration with clinicians from the Emergency and Admission Department of the Policlinico Umberto I in Rome, Italy. As a concrete example we consider the pulmonary embolism (PE) diagnosis guideline, as a selected fragment of the venous thromboembolism diagnosis and treatment guideline [26] adopted by clinicians. An adapted representation of the PE diagnosis guideline is shown in Fig. 7.

[1] See www.omg.org/spec/BPMN/.

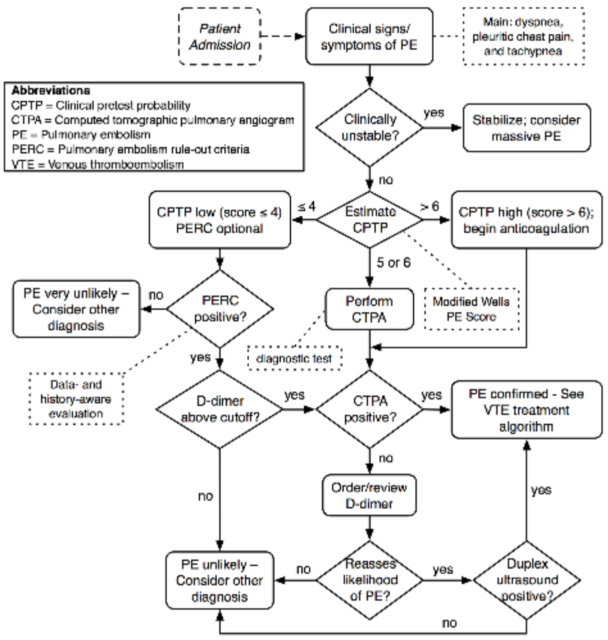
Patient Admission
Clinical signs/ symptoms of PE
Main: dyspnea, pleuritic chest pain, and tachypnea
Abbreviations
CPTP = Clinical pretest probability
CTPA = Computed tomographic pulmonary angiogram
PE = Pulmonary embolism
PERC = Pulmonary embolism rule-out criteria
VTE = Venous thromboembolism
Clinically unstable?
yes
Stabilize; consider massive PE
no
Estimate CPTP
≤ 4
CPTP low (score ≤ 4) PERC optional
> 6
CPTP high (score > 6); begin anticoagulation
5 or 6
Modified Wells PE Score
Perform CTPA
PE very unlikely – Consider other diagnosis
no
PERC positive?
yes
diagnostic test
Data- and history-aware evaluation
D-dimer above cutoff?
yes
CTPA positive?
yes
PE confirmed - See VTE treatment algorithm
no
Order/review D-dimer
no
yes
PE unlikely – Consider other diagnosis
no
Reassess likelihood of PE?
yes
Duplex ultrasound positive?
no

Fig. 7 Pulmonary Embolism (PE) diagnosis guideline

As a first step in the overall care process, patient registration, admission and triage activities performed by a clinic nurse result in the creation or retrieval of the patient's medical case file. Patient's clinical situation, recorded and documented in patient's medical history, is central and represent the shared, explicit knowledge that will drive the decision making and will evolve as a result of performed actions, made decisions and collected data. Initial patient assessment, performed by a responsible clinician mainly on the basis of her expertise and experience, may then lead to recognize clinical signs and symptoms of a suspected PE. At this stage, the clinician undertakes a targeted diagnostic procedure, by relying on the evidence-based guidance provided by the guideline, complemented with additional "knowledge layers" that include clinicians' basic medical knowledge [8], site-specific knowledge and patient-related information, so as to obtain a concrete patient-specific medical pathway. The guideline helps to understand and define an initial high-level structuring and scheduling of activities, and allows to identify the multidisciplinary team of medical experts that have to be involved (nurses, a radiologist, a sonographer, etc.). The overall common goal initially set by the clinical team is to confirm the initial embolism diagnosis, so that a suitable treatment plan can be defined, or to refute it, so that embolism can be ruled out and other diagnoses can be considered. To achieve this goal, diagnostic-therapeutic decisions (e.g., stabilize a clinically unstable patient), pharmacy actions and substance administrations (e.g., *begin anticoagulation*), and clinical evaluations (e.g., *clinically unstable?* and *reassess likelihood of PE?*) are combined, with the involvement of different health professionals of the medical staff. Clinical evaluations and decisions may require physician's checks and physical examinations (as in the evaluation of *clinical signs/symptoms of PE*), may rely on diagnostic tests/imaging to be scheduled, performed and evaluated (cf., *perform CTPA* and *CTPA positive?*), or may be based on clinical scores to be computed (as in the *estimate CPTP*) and other data- and history-aware evaluations or rule-out criteria (cf., *PERC positive?*) that require data request and gathering steps (e.g., by accessing the local health information system that stores medical records). In particular, data-aware conditions often act as eligibility criteria for the whole process (specific clinical signs/symptoms trigger the diagnosis steps), for specific decision-action steps (e.g., *CPTP high [score > 6]; begin anticoagulation*), and for moving from different stages of the overall care process (e.g., *CTPA positive* confirms the diagnosis and leads to the definition of a treatment plan).

Patient's medical case file is progressively updated with new clinical observations and with the results of each test, examination and activity. This shared knowledge enables the involved experts to evaluate and correlate different results to come up with new decisions, goals and plans of actions. While some decisions are driven by explicit measurable clinical parameters (such as score calculation results or thresholds defined for lab results), in other cases the process progression and evolution is determined by the clinician, even in contradiction with the guideline. For example, a clinician may still require a D-dimer evaluation in the case of a negative result for the PE rule-out criteria (PERC) that suggests to exclude PE from the diagnosis.

The main driver for process progression is not strictly given by activity completions, but rather by a combination of decision making and the availability and evolution of certain values for clinical data. Moreover, the work of clinical team members is largely interrupt- and event-driven. Changes in the operational context, variations in patient conditions and in other heterogeneous contextual information sources may occur unpredictably and at any time, requiring the ability to react to those changes and properly adapt and modify process behavior. A clinician may order a computed tomographic pulmonary angiogram (CTPA) for a clinically stable patient, but patient's status can suddenly change and become unstable. In such a case, the clinician has to reconsider the initial or current plan of actions and the goals, and immediately react to treat the emergency with the new goal of stabilizing the patient, without waiting for the CTPA results. However, as soon as the diagnostic results become available, the clinician may need to modify the undergoing care plan (e.g., treatment of a suspected massive pulmonary embolism) as a consequence of the new available knowledge, as she may discover, for example, that the selected treatment is contraindicated/incompatible with patient's status. This in turn requires the clinician to re-assesses and re-evaluate the situation and then act or plan the subsequent actions to be performed and goals to be achieved. Similarly, contextual site-specific factors such as a temporary unavailability of the medical device for performing the computed tomography scan (e.g., due to a fault or because in use for another patient with higher priority) may force the clinician to identify alternative examinations. Each decision may be grounded in the personal experience and expertise of each team member, or may be the result of collaborative decision-making among clinical team members. The gradual emergence of new knowledge influences the undergoing or planned actions, as well as the size and composition of the team, which may dynamically change over time. Some professionals are involved on-demand (e.g., laboratory technicians that perform the D-dimer test), whereas others are co-located and work as a team on the specific case. From a general perspective, healthcare processes reflect the combination of predictable and unpredictable elements and span over the entire spectrum of process management introduced in Section 2.2. Administrative and organizational steps, including patient registration/discharge and other activities in the diagnostic and treatment delivery stages (e.g., patient transfer, bookings and lab tests) are typically structured, stable and repetitive. Conversely, the diagnostic and therapeutic steps driven by clinical decision-making and medical case data are clearly knowledge-intensive activities that lead to loosely structured or unstructured processes.

3.3 Artful Processes

Knowledge workers such as managers, researchers, engineers, etc. typically carry out collaborative tasks, which require complex, rapid decisions among multiple possible strategies, in order to fulfill specific goals. Very often, they follow a process, although it is implicitly known only by themselves. In contrast to business processes, which are formal and standardized, often such processes are not even written down, let alone defined formally, and can vary from person to person, even when those involved are pursuing the same objective. Knowledge workers create these workflows "on the fly", to cope with many of the situations that arise in their daily work. Thus, while the framing process may be stable at an abstract level, the key details are not. Though frequently repeated, they are not exactly reproducible, even by their originators – since they are not written down – and can not be easily shared either. We denote these kinds of processes "artful" in the sense that there is an art to their execution. In many of them, it is primarily the content in each process instance – rather than the process itself – that determines the outcome. Furthermore, they are often developed or refined locally at the individual or small-team level. Thereby, the process cannot be easily separated from the specific people who perform it. They depend on the skills, experience, and judgement of the primary actors. This is what essentially characterizes artful processes within the class of KiPs: their behavior depends on contingencies and actors, therefore no predefined model exists.

As an example, we can consider the coordination of an international research project. Some deadlines are fixed, such as review meetings or annual budgeting reports, but the rest of the steps made to meet the project's requirements vary from case to case. The publication of a deliverable, the set-up of a possible demo, the outcome of a task-force or a work package depend on the objective of the projects, the partners involved, contingencies, and so forth.

Another example of artful process, by far more flexible, is the making of a scientific publication, i.e., the interplay of activities such as proposing, evaluating, writing, etc. behind the publication of a research paper. In that case, it is known that most of the activities are common, such as writing, proof-reading, commenting, etc. Also, the revision process is quite standardized. Although, now we are interested in the other perspective, i.e., documenting scientific work. From this viewpoint, barely any systematized procedures exist: depending on the type of paper (e.g., a survey, the presentation of experimental results, a position paper, etc.), its contents, the authors' preferences, etc., the steps made to come up with an article change in the order, in the assignment, in the interplay. The same author can participate in the composition of several paper, applying totally different strategies. Furthermore, if new detailed analysis on conducted experiments show interesting results, e.g., the process might suddenly change in order to move the main focus of a section, or even of the entire work.

On the whole, every instance of an artful process may behave differently, with respect to the actors involved and the contextual information that the process is enacted within. Hence, its model has to be flexible, allowing several alternatives at run-time execution, on one hand. On the other hand, it has to be designed to foresee unpredictable deviations from the expected workflow. Therefore, any model for artful processes must allow the actor to violate its rules at run-time.

Modeling an artful process does not necessarily mean that every aspect of the business context is covered. Some details can be ignored, because *(i)* the detailed information can vary from case to case, *(ii)* taking into account every information related to the context could lead to redundant, intrusive or misleading hypothesis, and *(iii)* some decisions have to be left to the intuition of the knowledge worker.

With respect to Section 2.2, artful processes range from the *loosely structured* to the *unstructured* categories. Referring back to the examples of the research project management and the writing of a research paper, the former may be thought of as a loosely structured artful process, the latter as an unstructured one.

The process model has not been formalized beforehand, as actors usually have neither an exact idea of its structure, nor the time to write it down. Hence, artful processes subvert the ordering of the typical business process life cycle (cf. Fig. 1). Mining the workflow can be considered the initial step, in order to draw an initial version of the model. It can be refined further according to the actors' feedback. Usually, process specifications are extracted out of event logs. Event logs, though, are useful to this extent when recorded by software applications that are meant to trace the steps they move within a given workflow. This is not necessarily the case for those tools that are typically used by knowledge workers: email clients, document writers, etc., are tailored to not more than a single task. Therefore, their scope is not broad enough to cover an entire process. However, being the artful processes' behavior initially unknown, rarely workflow-driven tools could be used.

Understanding artful processes involving knowledge workers can lead to valuable improvements in many scenarios. For instance, in *personal information management (PIM)*, i.e., how to organize people's own activities, contacts, etc., through the analysis of data that their software register on laptops, smartphones and tablets. Here, inferring artful processes in which a person is involved allows the system to be proactive and thus drive the user through its own tasks [10, 73]. Moreover, in *enterprise engineering*, where it is important to preserve more than just the actual documents making up the product data. Preserving the "soft knowledge" of the overall process (the so-called product life cycle) is of critical importance for knowledge-heavy industries. Hence, the idea is to take to the future not only the designs, but also the knowledge about processes, decision making, and people involved [35].

As a a final remark, we draw the attention to the fact that the mining step would naturally tend to raise the level of structuredness of artful processes: e.g., once the control flow of an (initially) unstructured artful process is discovered, the process can be considered as shifted into the category of the loosely structured.

4 Main Characteristics of KiPs

KiPs are inherently *people-centric*, as they are mainly performed by knowledge workers, i.e., autonomous decision makers with different backgrounds, expertise and experience [15]. Knowledge workers create, access, update and exploit different types of domain-specific knowledge to achieve intended goals performing activities that require decision making capabilities [7, 27]. Starting from KiPs' definitions available in literature (cf. Section 2.1) and on the basis of the application scenarios shown in Section 3, we have derived 8 key characteristics representative of KiPs. While in this section we provide our own definition of characteristics, their explanation is rooted in the research literature.

C1 Knowledge-driven: *The status and availability of data and knowledge objects drive human decision making and directly influence the flow of process actions and events.* Process-related knowledge evolves as a result of process progression and the occurrence of contextual events [49]. Explicit knowledge can be formalized and encoded in some form of knowledge base, so as to define knowledge objects, data, information and artifacts to be considered as part of process context and execution state. Implicit or tacit knowledge is linked to the capabilities and experience of process participants and is embedded in their work practices and decision choices [29]. Clinical decision making, for example, is highly knowledge-driven and depends on explicit knowledge sources (including medical knowledge and evidence, and case- and patient-specific information recorded in the medical case file) and on tacit knowledge, i.e., clinicians' expertise and experience. Furthermore, tacit knowledge and contingent information mainly determine the advancement of artful processes: their entire behavior changes on their basis.

C2 Collaboration-oriented: *Process creation, management and execution occurs in a collaborative multi-user environment, where human-centered and process-related knowledge is co-created, shared and transferred by and among process participants with different roles.* Process progression and completion often require a team-based approach. It depends on knowledge flows and transfers of data and knowledge objects between communicating process participants [9, 49, 50]. For instance, patient management involves a multidisciplinary team of co-located professionals with heterogeneous skills and expertise. Artful processes typically involve small teams of actors, who bring their competence into play in order to reach a shared objective.

C3 Unpredictable: *The exact activity, event and knowledge flow depends on situation- and context-specific elements that may not be known a priori, may change during process execution, and may vary over different process cases.* The knowledge worker is often not able to predetermine the overall process structure in terms of the activities to be executed and their ordering, the data and knowledge sources to be exploited and the roles and resources required for process progression and completion [58, 62, 71]. For example, the definition of a detailed emergency response plan ahead of time is just impossible if the specific information of the emergency has not yet emerged. Similarly, the definition of a clinical diagnostic procedure is highly patient-specific and its evolution is subject to unpredictable situations.

C4 Emergent: *The actual course of actions gradually emerges during process execution and is determined step by step, when more information is available.* Process participants continuously assess process progression and then act or plan the actions to be performed, depending on the process status and the available data and knowledge elements [58]. Each performed action and taken decision towards the achievement of a given goal has the effect of producing knowledge. It will be exploited for supporting subsequent decisions and determining the next goals to be achieved as well as the actions to execute [62, 71]. This is the case, for example, of an emergency response plan, whose overall structure may be initially unclear and is gradually determined step by step, through the collection of contextual information concerning the specific emergency. In a similar way, a clinician combines observation, reasoning and action to incrementally define the diagnostic or therapeutic steps, whose outcome drives the process progression.

C5 Goal-oriented: *The process evolves through a series of intermediate goals or milestones to be achieved.* These goals may be known a priori and predefined, or gradually defined as the result of acquired knowledge and previously achieved goals [9, 49]. For example, an emergency response plan is goal-oriented by nature, and the specific plan's objectives are often determined at run-time. Moreover, goals may be modified or invalidated as a consequence of occurring events, which had an impact on process state and execution context [58]. The achievement of a given goal, or the failure to achieve it, both represent domain-relevant knowledge that contributes to the decision making process.

C6 Event-driven: *Process progression is affected by the occurrence of different kinds of events that influence knowledge workers' decision making.* During process execution, process participants may have to react to different kinds of events, which can occur in any sequence. These events represent changes that affect process state, process-related data and knowledge, and process execution context and environment [9]. Changes in the process data as well as events related to the initiation and completion of activities may correspond to the achievement of process goals and may act as triggers for subsequent decision-action steps [16]. Contextual changes require to properly adapt and modify process behavior. For example, in emergency management scenarios, external events that come from the environment may prevent the correct enactment of a response plan, which needs to be dynamically adapted to the new contextual knowledge of the scenario.

C7 Constraint- and rule-driven: *Process participants may be influenced by or may have to comply with constraints and rules that drive actions performance and decision making.* Being a form of knowledge, rules and constraints can be either explicit and available in guidelines, policies and other sources of business rules, or implicit and thus embedded in participants' personal work practices [14]. Rules and constraints contribute to the definition of decision criteria and may act as eligibility paradigms for selecting the actions to be executed, as well as the knowledge and data sources to be exploited [62]. The structuredness tying the high flexibility of artful processes stems indeed from the need to comply to given constraints: for instance, the writing of a deliverable in a research project must end before that the deadline for its submission expires.

C8 Non repeatable: *The process instance undertaken to deal with a specific case or situation is hardly repeatable, i.e., different executions of the process vary from one another.* Emergency response plans, for example, are usually unique, as they reflect processes to be applied in a specific emergency situation. However, this does not exclude the possibility of predefining process fragments and templates to be selected and re-used in a context-dependent way. In addition, mining activities performed over the history of executed processes may contribute to the identification of action/event patterns and declarative knowledge (e.g., rules and constraints), which could be exploited to refine existing work practices and policies. Furthermore, it would foster the re-use of best practices and guidelines, and convert tacit knowledge in explicit knowledge objects [36]. This is the case of artful processes, where the ever-changing behavior of processes can lead to non-repeatable schemes as

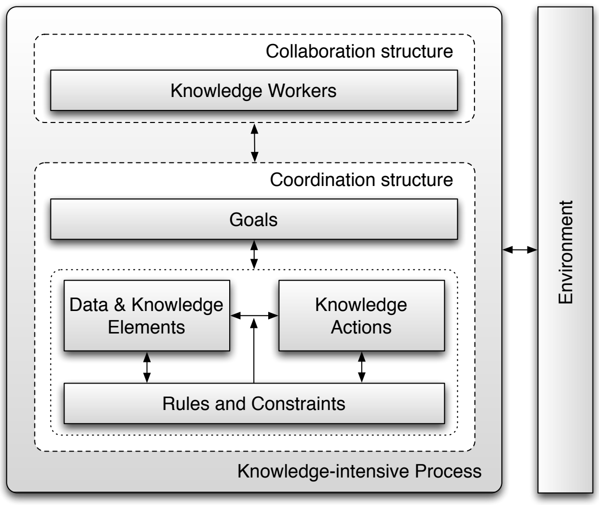

Fig. 8 Fundamental components of a KiP

a whole, nonetheless there is room for the identification of distinctive patterns and rules leading the execution.

5 General Requirements for KiPs

The analysis of real-world scenarios presented in Section 3 and the systematization of KiPs characteristics introduced in the previous section enable to identify the fundamental components of a collaborative KiP, as well as their interdependencies (cf. Fig. 8).

At the core level, it is possible to identify a tight integration of *data & knowledge elements* with *knowledge actions*. These components mutually influence each other: knowledge actions rely on the availability and content of data & knowledge elements, which in turn are affected by the performance of knowledge actions. The relations between different data & knowledge elements induce an information model that enables the flow of information to support actions' performance and decision making. This data-centric perspective emphasizes the need to capture and manage the structure, interactions and behavior of data & knowledge elements. The intra- and inter-dependencies between data & knowledge elements and knowledge actions are influenced and framed by *rules and constraints*, often related to guidelines and best practices. In particular, rules and constraints can define data and execution dependencies on knowledge actions and dictate their mandatory/optional nature. Similarly, they can express the aforementioned dependencies of knowledge tasks on data & knowledge elements, the impact of knowledge actions on the information model, and the effects of events and user decisions on the overall process structure. All the elements introduced so far directly relate to the specific *goals* to be achieved. Goals are mainly defined by *knowledge workers* and are gradually achieved as a result of actions' performance and data & knowledge evolution. The complex interdependencies among all these elements induce an overall coordination structure, coupled with the collaboration structure of knowledge workers. Both the coordination and collaboration structures dynamically change in relation to the actual context and *environment*, which impacts on goals, actions, data & knowledge elements and their interdependencies.

The identification of KiP components suggests that, in order to enable process-aware system support, the different interrelated elements have to be captured and managed along all the phases of the life cycle (cf. Fig 2). Therefore, we categorize the key requirements for KiP's support into 7 classes, that reflect the main components identified before. Moreover, according to the dynamics of the life cycle, the requirements in each category reflect the need to support the definition, evolution, monitoring and analysis of the corresponding component.

1. DATA

R1 Data modeling: *An information model including all relevant data manipulated by the process and their interrelationships* is required. Data can be more or less accurate, and may refer to different levels of abstraction, ranging from detailed properties provided by process variables to more aggregate information stored in data objects, which hold information structures pertinent to the global context.

R2 Late data modeling: The arising of new knowledge at run-time may involve the creation/modification of new/existing data. Therefore, a knowledge worker must be allowed to *add new data to the information model during the process enactment*, or to *alter or remove the existing ones.*

R3 Access to appropriate data: *All relevant data (such as contextual properties, emails, documents, etc.) must be accessible at any point of the process enactment to those participants having the required authorizations*, not only during the execution of a specific action.

R4 Synchronized access to shared data: *Different tasks/users may access and modify the same data concurrently at the same time, without the risk of affecting the integrity of data.* The consistency of data must be maintained during the process enactment.

2. KNOWLEDGE ACTIONS

R5 Represent data-driven actions: A KiP is characterized by actions whose enactment significantly depends on the evolution of the information model, so that purely data-driven process progression can be supported. It is therefore required that *knowledge actions are enriched with constraints (e.g., pre and post-conditions) defined on process data*, stating how data may constrain the action execution or may be affected after an action completion.

R6 Late actions modeling: To deal with the "emergent nature" of a KiP, *users must be allowed to add new knowledge actions to the process instance during its enactment, or to alter the existing ones.*

3. RULES AND CONSTRAINTS

R7 Formalize rules and constraints: In a KiP, the existence of policies, rules and regulations can influence the process structure and constrain its execution. To this end, *a user must be allowed to explicitly define constraints or business rules on process data.*

R8 Late constraints formalization: When new data or actions emerge during process enactment, *a knowledge worker must be allowed to add new constraints at run-time, or to alter the existing ones.*

4. GOALS

R9 Goals modeling: For a KiP, concrete goals may be created and their achievement may be associated to the result of acquired knowledge, i.e., when one or more data & knowledge elements assume a specific value determined by knowledge workers. Therefore, *a mechanism for representing one or more process goals defined on data and knowledge element is required.*

R10 Late goal modeling: During process enactment, new process goals may arise as a result of knowledge workers' decisions or due to the evolution of data & knowledge elements. *A knowledge worker must be allowed to associate new goals to a running process or to alter/remove existing goals that have became outdated.*

5. PROCESSES

R11 Support for different modeling styles: A KiP can be seen as a combination of knowledge entities (data, actions, etc.) having different degrees of structuredness. To possibly model any kind of KiP's schema, it is required to *provide the ability to select and combine various modeling alternatives.*

R12 Visibility of the process knowledge: *An aggregated perspective of data, actions, constraints and goals involved in a running process must be provided*, including their state as well as their interdependencies.

R13 Flexible process execution: A KiP is not dictated ahead of time but emerges as part of the collaboration and negotiation between the participants, which can decide to change the order of steps in the process and the type of information needed. *A knowledge worker must be able to "step back" or "jump forward", to re-execute previously performed actions, or to skip actions deemed unnecessary in a given instance.*

R14 Deal with unanticipated exceptions: A KiP is executed in environments that may change in unpredictable ways during its execution. The presence of unanticipated exceptions reflecting environmental changes or unexpected actions outcomes is common during a KiP's enactment. Hence, it is required to *catch unanticipated exceptions and provide mechanisms to generate the recovery procedure dealing with such exceptions*, which are either manual or completely automated, depending on the specific case.

R15 Migration of process instances: A KiP is often associated to environments, data and actions that evolve over time (due to changes in the business, in the technological environment, etc.). To maintain the running instances of a KiP aligned with the real-world specifications that emerge at run-time, *the migration of process instances into models compliant with new specifications* is crucial to support the execution of a KiP.

R16 Learning from event logs: A KiP must help an organization to learn from previous executed instances/cases. Therefore, it is required to *record event logs that trace the process progression and to provide mechanisms for discovering or improving the structure of a KiP, starting from the knowledge gathered from such logs.* A learning activity based on event logs may help to understand the impact of a KiP in real world, discover the KiP's process model, or check whether a pre-specified model is conformant with the event logs. It may also result in an improvement of the information model, in the definition of new actions, etc.

R17 Learning from data sources: The enactment of KiPs may have not been supported by a PMS in the past. However, there could be data reporting or tracing the execution of KiP, even though not formatted as event logs. Such data could consist in unstructured texts such as PDF documents, semi-structured texts such as email messages, structured texts such as CSV files, database entries, etc. In these circumstances, there could not be a direct match between the fulfillment of an activity and a record in a list of events, such as logs. Nonetheless, the capability of learning from the past should be guaranteed anyway. Therefore, it is required to *gather knowledge from heterogeneous data sources, in order to discover or improve the structure of a KiP.*

6. KNOWLEDGE WORKERS

R18 Knowledge workers' modeling: The *ability to define a resource model including multiple participants with multiple roles/capabilities* is fundamental for KiPs. Roles serve as a means of grouping knowledge workers with similar duties. Capabilities are used for specifying whether a knowledge worker provides the required skills to execute a specific action.

R19 Formalize interaction between knowledge workers: During the lifetime of a KiP, there is a range of involved knowledge workers who play different roles and collaborate during the process enactment. To this end, *mechanisms for defining structured or unstructured protocols that allow knowledge workers to communicate and collaborate are required.*

R20 Define knowledge workers' privileges: It is required to *define explicitly knowledge workers' privileges* for *(i)* specifying permissions for creating/altering/deleting data and knowledge elements *(ii)* avoiding that confidential information is made available to inappropriate knowledge workers.

R21 Late knowledge workers' modeling: Given the "emergent" nature of a KiP, it could be required to *insert new knowledge workers and their respective capabilities to the resource model at run-time*, *to alter capabilities of existing knowledge workers* or *to remove existing knowledge workers from the resource model.*

R22 Late privileges modeling: At run-time, it may be required to *add/remove/alter privileges associated to existing knowledge workers*, since new knowledge entities may arise during the KiP enactment.

R23 Capture knowledge workers' decisions: At run-time, decisions made by knowledge workers may affect the process progression (for example, the explicit selection between alternative execution paths) or the state of information model (for example, the direct manipulation of relevant data). To this extent, it is required to *capture knowledge workers' decisions at run-time and to associate their occurrence's impact on the process progression and on the information model.*

7. ENVIRONMENT

R24 Capture and model external events: An external event is a trigger coming from the environment that changes the state of the running process, by altering the value of data in the information model. Hence, it is required to *allow to explicitly represent external events coming from the environment and to associate their occurrence's impact on the information model.*

R25 External events late modeling: During process enactment, if a new external event (that was not previously captured) occurs, *a knowledge worker must be allowed to formalize it and to associate its occurrence impact on the information model.*

Requirement	Mainly induced by	Relevance for Em. Mgmt. Processes	Relevance for Healthcare Processes	Relevance for Artful Processes
R1	**C1 C2 C7**	high	high	high
R2	**C1 C2 C4 C7**	high	high	high
R3	**C1 C2 C7**	medium	high	medium
R4	**C1 C2 C3**	high	high	medium
R5	**C1 C3 C7**	high	high	high
R6	**C3 C4**	high	high	high
R7	**C7**	high	high	high
R8	**C3 C4 C7**	high	high	high
R9	**C5**	high	high	high
R10	**C3 C4 C5**	high	high	high
R11	**C1 C5 C6 C7**	high	medium	medium
R12	**C1 C2 C5 C7**	medium	high	high
R13	**C3 C4**	high	high	high
R14	**C3 C4 C6**	high	high	low
R15	**C2 C4 C6 C8**	low	high	medium
R16	**C1 C4 C8**	high	high	high
R17	**C1 C4 C8**	high	high	high
R18	**C1 C2**	high	high	high
R19	**C2**	medium	medium	medium
R20	**C1 C2**	medium	high	medium
R21	**C1 C2 C4**	high	high	high
R22	**C1 C2 C4**	low	high	medium
R23	**C1 C2 C4 C7**	high	high	high
R24	**C6**	high	high	medium
R25	**C4 C6**	high	high	medium

Table 1 KiPs requirements, the characteristics inducing them, and their relevance for the considered scenarios

Table 1 shows, for each requirement, the subset of characteristics (described in Section 4) relevant for the requirement itself and the requirements' relevance for the considered scenarios. For the interested readers, some specific research surveys have been realized for describing in detail the single aspects presented in our requirements. Among these, [56] evaluates several process modeling languages with respect to the role of data. Works [3] and [67] identify recurring, generic constructs in the control-flow and data perspectives, and present them in the form of control-flow and data patterns. [66] captures the various ways in which resources are represented and utilized in workflows, while works [68] and [74] suggest a set of adaptation patterns. With respect to the above works, our requirements provide a high-level overview of the features required for an effective support of KiPs. They are devised for being applied to a variety of application domains and for enabling full process life-cycle management.

6 Analysis of Contemporary Approaches

The requirements identified in Section 5 contribute to the definition of an evaluation framework to assess current system support for KiPs. In this section, the requirements framework is used to evaluate a selected subset of process-oriented systems and approaches. According to the selection criteria presented in Section 1.2, we evaluate 5 well known and 2 emerging approaches/systems coming from academia. The systems considered here are representative of the different process management paradigms that have emerged over the years: activity-centric imperative approaches for supporting flexible and adaptive processes (the YAWL and ADEPT2 systems), declarative approaches for supporting loosely structured processes (the Declare system), object-aware approaches (the PHILharmonicFlows system) and artifact-centric approaches (the ArtiFact system). In addition, we also analyze two recent research approaches resulting from our contribution (SmartPM and MailOfMine), which may complement or extend the current state of the art. Each system is briefly introduced[2] and then evaluated against the requirements:

YAWL. YAWL [32], Yet Another Workflow Language, is a modeling language grounded in workflow patterns [3] and in workflow nets [1]. It is based on a rich workflow definition language, capable of capturing all sorts of flow dependencies between tasks. The language is supported by a software system[3] (we consider here version 2.3.5) that includes a graphical editor, an execution engine and a task handler. The graphical editor offers visual support for the definition of process' control logic, variables and organizational resources. When a process is ready to be executed, the control is passed to the YAWL *Engine*, which is in charge of assigning tasks to proper participants.

ADEPT2. The ADEPT2 system[4] was introduced in [63] to support dynamic change of process models for unanticipated exceptions. ADEPT2 uses a block-structured modeling approach, and provides a meta-model for the integrated modeling of different process aspects including tasks, control and data flow, actor assignments and temporal constraints. ADEPT2 provides a graphical editor for modeling process schemes and creating participants assignment rules. New processes can be composed in a plug&play-like fashion and activities can be added/removed at run-time by drag & drop them from a pre-defined repository.

SmartPM. SmartPM [52, 54] (Smart Process Management) is a model and a prototype PMS featuring a set of techniques to automatically adapt processes at run-time. SmartPM provides a GUI-based tool that allows to explicitly represent data and knowledge elements associated with a process schema defined through the BPMN 2.0 notation. The process is then executed by a dedicated engine.[5] The adaptation features provided by SmartPM allow to adapt a running process if any unanticipated exception occurs at run-time, without the need to predefine any exception handling strategy at design-time. To accomplish this, SmartPM makes use of well-established techniques and frameworks from Artificial Intelligence, such as situation calculus [64], IndiGolog [17] and classical planning.

Declare. Declare [6] is a language and prototype[6] (we consider here version 2.2.0) that uses a constraint-based process modeling approach for the development and enactment of declarative models, effectively supporting the definition and execution of loosely-structured processes. It comprises three main tools: Declare Designer, Declare Framework and Declare Worklist. Specifically, Declare Designer allows users to model and create constraint models, define new constraint templates, and perform static verifications on created models. Declare Framework acts as an execution engine, provides support for the enactment and monitoring of constraint model instances, and allows ad-hoc changes of running instances. After that a process model is loaded in the Declare Framework, a user can use Declare Worklist, in order to instantiate new processes and execute active instances' tasks. Declare core is also a Java library. As such, it has been integrated with ProM [5], the Process Mining Toolkit,[7] for mining declarative workflows, and CPN Tools [77].[8]

PHILharmonicFlows. The PHILharmonicFlows framework and prototype[9] enables object-aware process management on the basis of a tight integration

[2] When applicable, we specify the evaluated version.

[3] `http://www.yawlfoundation.org/`

[4] `http://www.uni-ulm.de/en/in/dbis/research/projects/completed-projects/adept2.html`

[5] `http://www.dis.uniroma1.it/~smartpm`

[6] `http://www.win.tue.nl/declare/`

[7] `http://www.promtools.org/prom6/`

[8] `http://cpntools.org/start`

[9] `http://www.uni-ulm.de/en/in/dbis/research/projects/philharmonic-flows.html`

of processes, functions, data and users [39]. Process modeling and execution relies on two levels of granularity that cover object behavior (or life cycle) and object interactions. The framework, which comprises build- and run-time components, enables the definition of object types and object relations in a data model, while object behavior is expressed in terms of a process whose execution is driven by object attribute changes. The framework further provides support for coordinating the execution of related processes and the interactions of their corresponding objects.

ArtiFact – GSM. The business artifacts framework [13] provides a data-centric process management methodology focusing on business artifacts and their life cycles. The Guard-Stage-Milestone (GSM) meta-model [34] has emerged as a declarative framework for the specification of artifact life cycles and is supported by the ArtiFact system[10] v.1.0, which provides a modeling environment for creating artifact-centric models, an execution engine and a run-time environment.

MailOfMine. MailOfMine [19, 22] is an experimental prototype of a tool aimed at automatedly inferring previously unspecified artful process models out of semi-structured texts, contained in the email conversations exchanged among knowledge workers. To this extent, it exploits an interplay of text mining and process mining techniques. Its process modeling language is the same of Declare. At this stage of its implementation, it offers an effective and performant workflow discovery module, named MINERful [18], together with a prototype of workflow representation and monitoring panel, embedded in an email client. A dashboard presents the next activities to carry out, and an enriched email composition window helps the knowledge worker to write emails according to the artful processes running at the moment. The execution of tasks is meant to be possibly non-compliant to the process' constraints. In case, the running workflow structure is recalculated accordingly.

6.1 Systems Evaluation

The evaluation relies on the requirements classification framework introduced in the previous section and an overview of the evaluation results is presented in Table 2, which shows whether a system provides full (+), partial (∼) or no support (–) for each specific requirement. Sometimes a requirement may only be implicitly supported, but the feature is not part of the specification. Moreover, it may happen that some systems do not provide any native support for certain requirements, unless they are coupled or integrated with other systems or approaches. In such a case, we use the symbol "x/y" for indicating that the level of support for a given requirement rises from x to y.

Requirements	YAWL	ADEPT2	SmartPM	Declare	PHILharmonicFlows	ArtiFact – GSM	MailOfMine
Data							
R1	∼	∼	+	∼	+	+	–
R2	–	–	–	–	–	–	–
R3	–	–	–	∼	+	+	∼
R4	–	+	–/+	–	+	–	–
Knowledge Actions							
R5	∼	∼	∼	∼	+	+	–
R6	∼	+	∼	–/∼	–	–	–
Rules and Constraints							
R7	–	–	+	+	+	+	∼
R8	–	–	–	–/∼	–	–	–
Goals							
R9	–	–	–/+	–	–	+	–
R10	–	–	∼	–	–	–	–
Processes							
R11	–/+	–	+	–/∼	+	+	∼
R12	–	∼	–	∼	+	+	∼
R13	–	+	–	+	+	∼	+
R14	–/+	+	+	–	∼	–	–
R15	+	+	–	–/+	–	–	+
R16	–/+	–/+	–	–/+	–	–	+
R17	–	–	–	–/∼	–	–	+
Knowledge Workers							
R18	+	+	+	–	+	+	–
R19	–	–	–	–	–	–	–
R20	–	–	–	–	+	+	–
R21	+	–	–	–	–	–	–
R22	–	–	–	–	–	–	–
R23	∼	∼	∼	–	+	+	–
Environment							
R24	–/+	–	+	–	–	+	–
R25	–	–	–	–	–	–	–

Table 2 Evaluating some process oriented approaches against the requirements described in Section 5

6.1.1 Data

YAWL. During its development, YAWL has been focused on control flow patterns, and the role of data has not been formally specified in the language. However, at configuration time, the YAWL editor allows to define local and global variables - represented as XML structure - for constraining tasks execution and for evaluating branching conditions [**R1** ∼]. In YAWL, during process enactment, a knowledge worker can not access/modify/create any variable [**R2** –, **R3** –], and no policy exists that guarantees the integrity of a variable value during a synchronized access [**R4** –].

ADEPT2. ADEPT2 allows to represent data by means of global process variables, that are denoted as

[10] `http://sourceforge.net/projects/bizartifact/`

data elements. Specifically, data exchange between activities is realized through writing and reading data elements. In addition, a user can define complex *data objects* and associate them as input or output of an activity. Other aspects like the interrelations between data objects are not supported [**R1** ∼]. No further support is provided to data elements and objects; for example, ADEPT2 does not allow a user to define new data elements/objects at run-time [**R2** –] or to access to appropriate data at any point of the process enactment [**R3** –]. Conversely, ADEPT2 provides some kind of synchronization for writing the same data element at the same time. Specifically, for each write access to a data element, a new version of the respective data object is created and stored in a run-time database. This means that two concurrent activities writing on the same data element are actually writing on two different data objects [**R4** +].

SmartPM. In SmartPM, data are represented through some *atomic terms* that range over a set of *data objects* belonging to different *data types*. A user may define basic data types (e.g., boolean, integer, etc.) as well as complex data types (e.g., locations in a contextual scenario, etc.). A data object depicts an entity of interest (e.g., a specific location in the scenario). Atomic terms can be used to express properties involving domain objects, process participants and relations between them (e.g., an atomic term may be used to record the current location of a user). Argument types of a term (taken from the set of predefined data types) represent the finite domains over which the term is interpreted [**R1** +]. SmartPM does not allow a user to define new data objects at run-time [**R2** –] or to access to appropriate data at any point of the process enactment [**R3** –]. Furthermore, SmartPM is not natively able to guarantee synchronized access to shared data. However, if integrated with [51], SmartPM can automatically generate process models where concurrent branches are proven to be independent from each other (i.e., they cannot access to the same data at the same time) [**R4** –/+].

PHILharmonicFlows. PHILharmonicFlows relies on a relational data model for the definition of the information perspective. Data modeling is supported through the definition of object types, their attributes and relation types [**R1** +]. Based on the data model describing the domain-specific data objects, a corresponding data structure, which comprises a collection of object instances and their relations, dynamically evolves at run-time. While object instances may be created, deleted or updated at any point in time, the corresponding data model cannot be altered at run-time [**R2** –], i.e., it is not possible to add, delete or update object types, their attribute schema and relation types for a running process instance. From a user perspective, process participants are provided with both data- (overview of data objects and their attributes) and process-oriented (worklist-based) views. To support coordinated user access to object instances, PHILharmonicFlows allows to define role-based authorization policies for accessing, changing, creating and deleting object instances and their attributes. Authorization settings are related to the dynamic behavior of object instances. The behaviors and life cycle of object instances are expressed as "micro processes", i.e., state charts whose transitions are driven by object attribute changes. Each micro-process state comprises several micro steps, each representing an atomic action as a mandatory write access on a particular object attribute (or object relation) of the respective object instance. At run-time, when an object instance is created, a corresponding micro-process instance is created. At any point during micro-process execution, only one state is enabled, and a micro-process instance in a particular state may only proceed if (specific) values are assigned to the attributes associated with this state. The assignment of attribute values occur through form-based activities, which can be executed only by authorized users. At the micro-process level, state types are associated with user roles, and automatically generated authorization tables are used at run-time to ensure that data access is constrained by user privileges. The possibility of defining optional data permissions for user roles not associated to a state type allows users to access process relevant data at any point in time, so that optional access to data is enabled asynchronously to process execution [**R3** +]. The overall coordination of the processing of object instances with user involvement ensures that data consistency is preserved. Although a micro-process instance is always in one micro state, concurrent executions involving multiple users operating on a same object instance are supported through concurrent data access mechanisms [**R4** +].

ArtiFact – GSM. A business artifact includes business-relevant data about a business entity, along with information about the life cycle that the entity moves through. It encompasses the key stages of the processing of the entity and how they are (or might be) sequenced. The *information model* of a business artifact type holds the information needed for completing business process executions in a hierarchical description. It is connected to a given business entity and includes references to related artifacts [**R1** +]. *Data* and *status* attributes are part of it. The former hold domain-specific information, the latter contain information about the progress of an artifact instance. A predefined information model cannot be altered at run-time, i.e., changes over the data schema are not supported [**R2** –]. The information related to each artifacts determines altogether the run-time state of a business process. By exploiting a query-based mechanism, authorized users are allowed to access and manipulate artifact instances at any point in time using a predefined view. Such view is dynamically assigned to them according to their role in the business process and on the basis of an authorization model [**R3** +]. Artifact instances can be created and destroyed over time, and both attributes and their values can be created, updated, or deleted by the services/users in the process environment, according to the corresponding data schema. However, concurrency control mechanisms are not provided in the case of concurrent tasks affecting the same data attributes [**R4** –].

Declare. Declare Designer offers the opportunity to specify some basic information about data, given as input or output for an activity. Data are represented as variables either of boolean, numeric or string type [**R1** ∼]. The collaborative aspect of multiple users enacting a process is not considered in Declare yet: therefore, no policy on synchronized access to data is taken into account [**R4** –]. Due to the same reason, the evolution of involved variables can be monitored by Declare Worklist, but no mechanism to control/grant the access is available [**R3** ∼]. At the moment, late data modeling is not a feature that Declare provides [**R2** –].

MailOfMine. MailOfMine comes bundled with an email client. Therefore, the access to knowledge items is guaranteed as long as they are attached to email messages [**R3** ∼], since the connection between email messages and performed activities is shown in the system. Although, no explicit modeling or access control of data is currently given [**R1** –, **R2** –, **R4** –].

6.1.2 Knowledge Actions

YAWL. In YAWL, tasks execution is based on input and output parameters defined over variables (access to variables is handled through XPath and XQuery), so that process progression is affected by both control flow and data [**R5** ∼]. At design-time, it is possible to associate one or more *placeholders* to YAWL activities in order to defer their modeling at run-time. Late modeling is therefore supported by YAWL, but only in some specific points of the process [**R6** ∼].

ADEPT2. The execution of an ADEPT2 model follows informal token semantics and is driven by a mixture of control flow and data aspects. Each input/output parameter of a particular activity is mapped to exactly one data element through a *data edge*. Data edges either represent a read or a write access of an activity to a data element [**R5** ∼]. At run-time, if compared with YAWL, ADEPT2 supports a more flexible version of *late modeling*, that allows to create/alter new/exisisting tasks at run-time and to insert them at any stage of the process [**R6** +].

SmartPM. In SmartPM, process tasks are annotated at design-time with pre-conditions (to constrain the task assignment) and desired effects, defined as logical conditions over atomic terms [**R5** ∼]. At run-time, late modeling of process activities is allowed only in presence of a catched exception and only if the adaptation algorithm provided by SmartPM is not able to find any recovery procedure for the specific exception. If so, late modeling is limited to the manual insertion of additional tasks in the point of the process where the deviation has been identified [**R6** ∼].

PHILharmonicFlows. In PHILharmonicFlows, the behavior of a data object is expressed as a micro process in terms of possible states and transitions. The definition of which object attribute values must be set to exit from a micro-process state contributes to the definition of data-driven activities [**R5** +]. As a consequence, the progress of an object instance is driven by changes of the corresponding attributes, enabling a data-driven process execution. This holds for form-based activities allowing users to set object attributes, and for black-box activities defined for integrating arbitrary application components. As process actions are data-driven, the support for late action modeling is related to the possibility of performing changes at run-time over the domain data model. However, the lack of support for late data modeling (cf. **R2**) makes late action modeling not explicitly supported [**R6** –].

ArtiFact – GSM. The data-centric nature of GSM is given by the definition of artifact life cycles in terms of stages, each associated with one or more milestones and guards. A *stage* identifies a cluster of activities related to an artifact instance. Composite stages enable the nesting of (sub-)stages, whereas atomic stages contain tasks that consist in the execution of specific activities or in the invocation of services that operate on the information model. Stages are controlled by the associated *guards*, i.e., expressions that determine whether a stage becomes active or open, so that sub-stages can be considered or the corresponding tasks can be executed. Similarly, the closing of stages is controlled through *milestones*, i.e., expressions that represent business-relevant operational objectives (at different levels of granularity) that can be achieved or invalidated. Guards and milestones determine the progress of artifact instances. Expressions for guards and milestones (referred to as *sentries*), consist of a triggering event and/or a condition, and have the form of Event-Condition-Action (ECA) rules. The triggering events may be incoming (external) or internal, and both the events and the conditions may refer to the information model of the artifact instance under consideration, to other artifact instances in the overall artifact system, and to the status of stages and milestones. Task are thus inherently data-driven, as their activation is constrained by data-based conditions. Similarly, task executions produce output data that is written back into the artifact instance information model [**R5** +]. Concerning late action modeling, no support is provided for altering tasks in a running process instance [**R6** –].

Declare. Declare permits the specification of basic conditions on variables for constraints. They act as preconditions, in the sense that if and only if a user-defined propositional formula on data holds true at run-time, then the constraint is triggered. Although, the formalization of a specification language as well as the full implementation is still an ongoing work (cf. [57]) [**R5** ∼]. The specification of activities, data and conditions can only be given at design-time. However, Schunselaar et al. [70] have proposed an extension, named Configurable Declare, allowing the users to hide events, i.e., do not monitor the execution of given activities [**R6** –/∼].

MailOfMine. At the current stage of implementation, MailOfMine does not provide either any formalization of pre- or post-conditions on data, nor the possibility to alter at run-time the list of actions that the process is constituted of [**R5** –, **R6** –].

6.1.3 Rules and Constraints

YAWL. Given its limited support to data, YAWL does not provide any mechanism for defining constraints and business rules on process data [**R7** –, **R8** –].

ADEPT2. In ADEPT2, the general data support is limited to the linkage of data elements to activities as input or output, and no further constraints on data elements may be defined [**R7** –, **R8** –].

SmartPM. SmartPM allows to define at design-time rules and constraints based on atomic terms through the so called *abbreviations.* Abbreviations, unlike atomic terms, are not directly affected by actions (i.e., they cannot appear as action effects). However, similarly to atomic terms, their value may vary after each task completion, as they depend on a combination of atomic terms that are possibly modified by actions completion [**R7** +, **R8** –].

PHILharmonicFlows. The rich data model of PHILharmonicFlows naturally enables the definition of different constraint types [**R7** +]. Minimum and maximum cardinalities can be specified for relation types among object types, and for each object type exactly one key attribute type is defined. At run-time, object instances (and corresponding micro-process instances) can be dynamically instantiated according to the cardinality constraints defined in the data model. These constraints have a direct impact on process progression: to ensure that a run-time data structure meets the cardinality constraints in the model, specific data-creation activities are automatically assigned to authorized users, in order to satisfy minimum cardinality. Object creation is disabled when a maximum cardinality is satisfied. Similarly, different kinds of synchronization constraints can be defined for coordinating the interactions between the object instances of the same or different object types. However, as for data and actions, at run-time no modeled constraints can be modified and no new constraint can be introduced [**R8** –].

ArtiFact – GSM. Although an artifact is supposed to be self-contained, it can contain references to other artifacts. Possible relationships among artifact types can be complemented with static constraints, such as key, multiplicity, disjointness or inclusion constraints. The identification of business artifacts is coupled with the definition of their life cycles, that identify business-relevant phases in the possible evolution of the artifact instances. The life cycle of an artifact type is a specification of a set of dynamic constraints on the allowed sequencing of the phases traversed by its instances, which describe how an artifact can evolve over time. The overall evolution of an artifact instance is controlled by ECA-like rules. Since ECA rules can refer to incoming events, internal events and data/status attributes in the information model, they provide a direct mechanism for representing business rules and constraints [**R7** +]. Both static and dynamic constraints cannot be changed or created for a running instance [**R8** –].

Declare. Declare Designer offers support to the specification of declarative models for workflows, constituted by sets of constraints, i.e., temporal rules [**R7** +]. Declare is designed to offer the user the opportunity to either specify constraints which belong to predefined constraint templates, or define new constraints by means of LTL formulae [60]. The aforementioned extension of Schunselaar et al. [70] lets Declare allow the users to omit constraints from the specification [**R8** –/∼].

MailOfMine. MailOfMine is made to discover declarative processes that lay behind the exchange of email messages of knowledge workers. The output is a model complying to the specification of Declare, and it is based on constraints. Although, no option offering the user the opportunity to specify some constraints from scratch is provided [**R7** ∼]. Currently, users cannot change the discovered declarative model either [**R8** –].

6.1.4 Goals

YAWL. In YAWL, given a specific process model, the achievement of process goals is associated to the correct completion of one of its process instances. However, YAWL does not allow to define concretely any goal based on process data [**R9** –, **R10** –].

ADEPT2. ADEPT2, like YAWL, does not allow to formalize concretely any process goal defined on data elements/objects [**R9** –, **R10** –].

SmartPM. If coupled with [51], SmartPM allows to formalize a process goal as a conjunction of atomic terms to make true through the execution of a process. A concrete goal can be thus used for automating the generation of a process model [**R9** –/+]. At run-time, new goal conditions may arise for driving the generation of recovery procedures, but their achievement does not affect process progression [**R9** ∼].

PHILharmonicFlows. In PHILharmonicFlows, no specific support is provided for explicitly representing process goals [**R9** –, **R10** –].

ArtiFact – GSM. In the context of an artifact life cycle, explicit support for representing goals is provided through the definition of *milestones* [**R9** +], i.e., business-relevant operational objectives that can be achieved or invalidated. The artifact information model includes all the data needed to *(i)* capture business process goals, and *(ii)* evaluate whether these goals are achieved. The achievement (or invalidation) of a milestone directly contributes to process progression, as it is considered as an internal event that possibly determines the opening of stages, the achievement of other milestones, etc. Milestones are pre-determined and no support is provided for altering them at run-time [**R10** –].

Declare. No direct support to the definition of process goals is currently provided in Declare [**R9** –, **R10** –].

MailOfMine. MailOfMine does not offer any facility to specify process goals [**R9** –, **R10** –].

6.1.5 Process

YAWL. The YAWL system is characterized by a service-oriented approach that makes the system easily extendable and provides direct support for implementing the *flexibility as a service approach* [4]. Different services may implement the corresponding YAWL activities using different workflow languages. With this approach, different styles of modeling may be mixed and nested in any way appropriate. For example, YAWL may be easily combined with the Declare system [59] (see later) to support arbitrary mixtures of loosely-structured and highly-structured processes [**R11** –/+]. YAWL does not provide an aggregated view of the process knowledge at run-time (all data dependencies are hidden and not explicitly shown) [**R12** –] and does not allow to deviate from the execution flow prescribed at design-time [**R13** –]. At run-time, for each exception that can be anticipated, it is possible to define an exception handling process, named *exlet*, which includes a number of exception handling primitives and one or more compensatory processes in the form of *worklets* (i.e., self-contained YAWL specifications executed as a replacement for a work item or as compensatory processes). However, YAWL does not provide natively any support for unanticipated exceptions. A recent approach (named the Planlets approach [53]) has been devised for enriching the YAWL architecture with mechanisms that deal with unanticipated exceptions [**R14** –/+]. YAWL provides also some form of *evolutionary change* caused by the modification of a process model. Specifically, when a process model is modified at run-time, all its running instances are aborted, compensated and restarted or migrated to the new process model, while the new instances are created according to the new process model [**R15** +]. Finally, even if YAWL does not directly provide any mechanism to learn from previous executed process instances, it allows to create log entries whenever an activity is enabled, started, completed, or canceled. Such event logs are converted in the so-called Mining XML (MXML) log format, which can be used for post-execution analysis in the ProM environment [5], one of the most used and well-known process mining toolkits available [**R16** –/+]. Although, no support to gather knowledge from heterogeneous data sources is provided [**R17** –].

ADEPT2. ADEPT2 allows to create process models describing the control flow for the process activities as well as the data flow between them. The collection of data elements and data edges constitutes the *data flow schema*. For each process instance and its data flow schema, the current execution of data edges is used to derive the state of the process and present it to the user [**R12** ∼]. While ADEPT2 does not allow to combine different modeling styles [**R11** –], it is one of the few PMSs that provide integrated support for dynamic structural process changes at different levels. Firstly, in ADEPT2 a process instance can deviate at run-time from the execution path prescribed by the original process without altering its process model [**R13** +]. Secondly, ADEPT2 is able to support the handling of unanticipated exceptions, by enabling different kinds of ad-hoc deviations from the process instance at run-time, according to the structural process change patterns defined in [74] [**R14** +]. Notice that the associated recovery procedure must be built manually by a process designer at run-time. Thirdly, ADEPT2 supports dynamic evolution of process schema and associated instances, i.e., changes to the process schema are propagated to already running process instances [**R15** +] by guaranteeing the compliance of migrated process instances with the new schema version. In ADEPT2 no native support is provided to learn from previous executed process instances. However, ADEPT2 allows for recording change logs in addition to traditional execution logs, by obtaining an *abstract change process*. It reflects all changes applied to the instances of a particular process type and may serve as basis for deriving process optimizations in the future. In [30], it is shown how to integrate the process mining framework ProM [5] with ADEPT2 [**R16** –/+]. As in YAWL, no support to gather knowledge from heterogeneous data sources is provided [**R17** –].

SmartPM. The definition of process models in SmartPM involves combining imperative constructs (e.g., control flows) with declarative elements used for associating atomic terms to tasks and to create complex constraints based on atomic terms [**R11** +]. The dynamic world of SmartPM is modeled as progressing through a series of *situations*. Each situation is the result of various tasks being performed so far. Atomic terms may be thought of as “properties” of the world whose values may vary across situations. However, the current version of SmartPM does not allow to visualize explicitly the status of the knowledge during the process progress [**R12** –]. SmartPM provides mechanisms for adapting process schemes that require no predefined handlers. Specifically, adaptation in SmartPM can be seen as a way to reduce the gap between the *expected reality*, i.e., the (idealized) model of reality that is used to reason, and the *physical reality*, i.e., the real world with the actual values of conditions and outcomes. At run-time, the physical reality can be invalidated due to task failures or external events, preventing the process progression. A recovery procedure is needed if the two realities are misaligned from each other. The adaptation algorithm deployed in SmartPM synthesizes a linear process (i.e., a process consisting of a sequence of tasks) that “repairs” the original process by removing such gap [**R14** +]. Currently, SmartPM is only able to change a process instance at run-time in case of exception handling [**R13** –], and no strategies for process evolution and mining have been still implemented in the system [**R15** –, **R16** –, **R17** –].

PHILharmonicFlows. The framework provides support for coordinating the execution of related micro processes and the interactions of their corresponding objects through the definition of “macro processes” that model multi-object interactions. A macro process refers to object instances of the data structure and consists of macro steps and macro transitions between them. A macro step refers to a particular object type and its state, and macro steps may be connected using macro transitions to express object interactions. Different kinds of synchronization constraints may be defined for coordinating the interactions between the object instances, including parallel and alternative execution paths. Declarative and procedural modeling styles can thus be combined [**R11** +]. In addition PHILharmonicFlows allows differentiating at run-time between a data- and an activity-driven execution paradigm.

Process-related knowledge is mainly captured in the data model. As a consequence, at run-time the overall state of a process is given by the actual status of object instances and their relations or interactions. The data model thus provides the basis for dynamically creating data-oriented views that reflect process status, as an aggregated view on existing object instances and their interdependencies. In particular, overview tables can be used to visualize, for each object type, its corresponding object instances. As user modeling is integrated into the data model, the status of process participants is easily accessible as well. This explicit visibility of process-related knowledge [**R12** +] supports user decision making, as overview tables can be used to initiate activities on selected object instance.

According to the data-driven process progression mechanisms, users can arbitrarily instantiate objects (and their corresponding micro processes) when needed. Users can also skip, redo or re-initialize activities [**R13** +]. Although this increases process flexibility, there may still be the need to deal with exceptions and perform ad-hoc changes over running instances. Currently, exception handling capabilities are limited and specific techniques to ensure a correct execution of micro and macro processes at run-time need to be defined [**R14** ∼]. As a form of exception handling, PHILharmonicFlows includes a detection algorithm for identifying deadlocks that prevent the data structure to evolve, and assists users in resolving them. Moreover, different kinds of exception handlers can be used to deal with so-called bypassed micro process instances (cf. [62] for the details). Similarly, the challenges related to schema evolution and instance migration are under investigation but no support is currently provided [**R15** –]. Process mining and analysis are currently not supported [**R16** –, **R17** –].

ArtiFact – GSM. The declarative modeling approach of GSM provides support for different modeling styles, as ECA rules can be used to both reproduce procedural patterns and define flexible execution behaviors [**R11** +]. At run-time, process execution does not follow a predefined order, as it is driven by the availability of data elements and the actual status of artifact instances. Users can influence task executions by generating events that trigger the opening of a stage and induce the activation of the enclosed tasks. Activity-repetition rules are not directly supported, though. The lack of an explicit task life cycle prevents the possibility of skipping activities, suspend them, etc [**R13** ∼]. The overall approach combines process control, data flows and human-centered knowledge in a unified view, that facilitates the visibility and monitoring of process progress. At any point in time the run-time state of a business process is determined by the *snapshot* of all artifacts: this includes an aggregated view of interrelated artifact instances and their data. It shows the actual status of stages (open/closed) and enclosed tasks, as well as the achievement of milestones [**R12** +]. Intrinsic flexibility reduces the need for adaptation and exception handling, but issues related to process adaptation and evolution are not explicitly addressed [**R14** –, **R15** –]. Similarly, learning and discovery mechanisms are not yet supported [**R16** –, **R17** –].

Declare. Currently, Declare is being integrated with CPN Tools [77], a well known tool for the design and validation of Colored Petri Nets. The claimed objective is to provide a hybrid declarative/imperative modeling [**R11** –/∼]. The Declare Worklist by itself lets the user enact and monitor the current execution of a process instance. Each activity in the Worklist contains “start” (play) and “complete” (stop) icons, that indicate whether users can begin and, resp., end the activity. The set of verified constraints is shown and updated at run-time [**R12** ∼] and the next executable activities are suggested accordingly. Some constraints can be violated, when they are specified as non-mandatory [**R13** +]. Declare Maps Miner [45] is a ProM plug-in for mining declarative processes out of event logs, based on the Declare framework. It is able to both discover new workflows and repair existing maps – i.e., it can add or remove constraints in order to update the model to a new version that fits the log [**R16** –/+]. Both functionalities are meant to work with XES or MXML-formatted event logs. ProM provides the opportunity to convert CSV files into XES/MXML-formatted event logs, though, thus partially supporting the discovery from heterogeneous data sources [**R17** –/∼]. The aforementioned Configurable Declare extension is meant to allow the user to hide events or remove constraints with run-time instances [**R15** –/+]. No exception handling paradigm is currently implemented in Declare [**R14** –].

MailOfMine. MailOfMine is designed to discovery declarative process models out of previous email conversations [**R17** +]. MailOfMine offers the opportunity to have a process mining tool embedded in an email client. Therefore, it allows to either visualize the process model, or keep it hidden behind proper suggestions during the composition of new email messages. Both a run-time representation of the current process instance and a static visualization of the model are provided [22] [**R12** ∼]. In particular, the static visualization offers two scopes on the representation of the model: one involves all the activities at once, the other focuses on the constraints regarding a single activity at a time [20]. However, the modeling approach is still unique and referred to the Declare standard templates [**R11** ∼]. MailOfMine guarantees a high flexibility w.r.t. the process model: the knowledge worker can violate constraints during the enactment of the workflow [**R13** +]. The run-time execution itself is recorded and analyzed [**R16** +]. Hence, in case of deviations from the expected behavior, the process model is updated accordingly [**R15** +]. MailOfMine cannot handle exceptions [**R14** –].

6.1.6 Knowledge Workers

YAWL. YAWL offers comprehensive support for the resource patterns [66], and the language allows to define process participants having multiple roles and capabilities, both at design-time [**R18** +] and run-time

[**R21** +]. As YAWL provides the ability to incorporate alternative execution paths within a process model at design-time, explicit users decision at run-time are limited to the selection of the most appropriate execution path for each process instance [**R23** ∼]. While YAWL provides a number of features for customizing privileges involving process participants and tasks, it does not provide any mechanism to define privileges involving participants and data/process variables [**R20** –, **R22** –]. Finally, the language currently does not support the definition of collaboration protocols between process participants [**R19** –].

ADEPT2. In ADEPT2, multiple participants and roles can be defined at design-time [**R18** +], but the resource model cannot be altered at run-time during a process enactment [**R21** –]. With ADEPT2, participants assignment rules may be created, but no support for participants collaboration through exchange of data elements/objects is provided [**R19** –], and it does not allow to define any explicit privilege to specify if a user may (or may not) interact with specific data elements/objects [**R20** –, **R22** –]. Conversely, ADEPT2 (like YAWL) provides a basic support to explicit users decisions, which is limited to the run-time selection between alternative execution traces [**R23** ∼].

SmartPM. SmartPM allows to define multiple participants with different roles and capabilities at design-time [**R18** +, **R21** –], but the collaboration aspects between them is not supported [**R19** –] as well as all the aspects related to the definition of privileges involving participants and atomic terms [**R20** –, **R22** –]. Finally, SmartPM provides a basic support to explicit users decisions, which is limited to the run-time selection between alternative execution traces [**R23** ∼].

PHILharmonicFlows. In PHILharmonicFlows the organizational model that defines users, roles and capabilities is integrated into the data model. User roles are modeled as object types, denoted as *user types*. Additional user roles are induced by possible relations between user types and object types [**R18** +]. The integrated modeling of data and users enables the definition of complex authorization and permission schemes. The system supports the generation and definition of authorization tables that take into account user roles, object types and their possible states (according with the corresponding micro processes). This allows restricting data access of a particular user to a subset of the instances of an object type, as well as defining read/write access control policies over data attributes [**R20** +]. Although the process structure enables a coordinated multi-user execution, no explicit communication and collaboration features are provided [**R19** –]. However, user decisions contribute to process progression. Process progression is the result of a combination of data evolution and explicit user decisions. Users' decisions affect object instantiations as well as the progress of an object instance. Moreover, users may read or write the attributes of an object instance asynchronously with respect to the execution of the corresponding micro-process instance [**R23** +]. As user types are basically data types, the lack of support for late data modeling does not allow to add, change or delete user types and their relations with object types at run-time [**R21** –]. Similarly, no support is currently provided to alter permissions for accessing processes, data, or authorization tables [**R22** –].

ArtiFact – GSM. Multi-user support is directly provided, and constraints can be defined to restrict data visibility and enable role-based task executions [**R18** +]. In particular, access control and authorization models can be defined [**R20** +]. The access control model describes two types of access rights: *(i)* data access rights, defining which (and under which conditions) data attributes of an artifact can be read or written; and *(ii)* service access rights, defining which artifact services can be invoked, by who, and under what conditions. The authorization model serves as a basis for defining role-specific views, as previously described. However, no support is provided for late modeling of users and their privileges [**R21** –, **R22** –]. User decisions explicitly drive process progression and data evolution. Users can generate events having an impact on the data attributes of an artifact instance, so as to trigger specific stages and execute the enclosed tasks. According to the declarative, rule-driven model, at run-time multiple execution options, alternative decisions and applicable actions may be available, depending on which services or events are contextually eligible during the life-cycle execution of an artifact instance. While the system provides structured guidance and control, the exact course of action can be the result of user decisions [**R23** +]. Although the overall process structure enables a coordinated multi-user execution, no explicit communication and collaboration facilities are provided [**R19** –].

Declare. The figure of process actors, with roles, capabilities and access grants is not considered by Declare [**R18** –, **R19** –, **R20** –, **R21** –, **R22** –, **R23** –].

MailOfMine. MailOfMine comes bundled with an email client. As such, it registers the name and email address of those knowledge workers that are usually involved in communications via email. However, it does not allow to model the interaction of knowledge workers with the process [**R18** –, **R19** –, **R20** –, **R21** –, **R22** –, **R23** –].

6.1.7 Environment

YAWL. In YAWL, process progression depends on tasks completion and on the occurrence of internal events, while external events are not natively supported. However, the integration of YAWL with the Planlets approach [53] allows to explicitly formalize external events at design-time and to specify how their occurrence at run-time may affect the value of process variables [**R24** –/+]. Late modeling of external event is not provided [**R25** –].

ADEPT2. While ADEPT2 allows to model the contextual properties of an external environment through data elements/objects, it does not provide any support to model and capture external events coming from the environment [**R24** –, **R25** –].

SmartPM. In SmartPM, the occurrence of events coming from the external environment may put at risk process progression. The list of external events and their occurrence's impact on atomic terms is specified at design-time [**R24** +], while no mechanism for defining external events at run-time is provided [**R25** –].

PHILharmonicFlows. While the evolution of a process structure can be related to implicit internal events (state transitions, data changes, etc.), no explicit support is provided for event modeling and event-driven execution [**R24** –, **R25** –].

ArtiFact – GSM. As a consequence of the ECA rules that characterize artifact life cycles, GSM is strongly event- and data-driven, as artifact instances move through their life cycles as the result of events that, when processed, may result in a series of guards becoming true and/or milestones changing value, along with stages becoming open and/or closed. Therefore, event modeling is explicitly supported to capture the interaction between artifact instances and the environment, as well as to model direct user requests (artifact creations, explicit decisions, etc.). Events carry a payload and, when processed, have a direct impact on the attributes of artifact instances, as the payload content is incorporated in the information model [**R24** +]. The overall event model is defined at design-time and no late modeling is supported at run-time [**R25** –].

Declare. Currently, Declare does not manage events which are external w.r.t. the activities comprised in the process specification [**R24** –, **R25** –].

MailOfMine. MailOfMine does not capture events that are not explicitly defined by the user as domain-related. Therefore, only those events that are meant to represent the execution of activities in the process enactment are considered. External events are thus not contemplated [**R24** –, **R25** –].

7 Discussion

The results of the evaluation conducted in Section 6 underline that none of the process-aware approaches and systems we analyzed is able to provide a complete support to the requirements described in Section 5. It is clear that KiPs reveal some challenging characteristics (such as collaboration-orientation, low predictability, evolvement during process enactment) that pose serious problems for their support by means of the existing process-oriented systems. Indeed, while BPM technology is considered mature enough for supporting organizational and administrative processes, process-oriented methodologies show limitations and pitfalls when dealing with the collaborative and emergent nature of KiPs.

Imperative approaches like YAWL and ADEPT2 support data in a primitive way, by focusing mainly on the control-flow perspective of the process. As a consequence, the two systems do not provide any specific feature that allows knowledge workers to access and modify the information model directly. Data access is possible only after an activity is completed, according to defined control flow. Therefore, data can not be accessed independently from process execution. Moreover, goals and external events are not part of their specification. Similar limitations hold for SmartPM, even if it allows for a more detailed definition of constraints over data, and provides basic primitives for modeling and capturing goals and external events. As for aspects related to dynamic change and process flexibility at run-time, YAWL provides only partial support for adapting and evolving a process (in case of expected exceptions). On the other hand, ADEPT2 provides a more complete support for process change and evolution in case of unanticipated exceptions, both at process schema and instance level. The strategy used for devising a recovery procedure is manual, though, and requires the human intervention at run-time. However, for a KiP there is no clear correlation between a change in the context and the corresponding process changes. Therefore, we think that the approach proposed by SmartPM represents a valuable contribution for supporting the enactment phase of a KiP, as it provides adaptation policies that do not require any manual intervention at run-time for the generation of a recovery procedure dealing with unanticipated exceptions.

Recent and ongoing works show that declarative languages such as Declare can be effectively used to increase the degree of flexibility, as resulting models have no rigid control-flow structure. Nonetheless, they still provide a good level of support. Such intrinsic peculiarity makes the declarative approach suitable to artful processes in particular. Artful processes indeed tend to be loosely structured and highly subject to change. In the field of process management, declarative modeling approaches are relatively new and less established than the imperative ones. However, the perspective of the control flow has been predominantly taken into account so far. Still very little consideration has been given to the integration of actors and roles into the definition of declarative processes. As a result, the tools supporting declarative workflows do not provide any facility to this extent. This is a defect, though, as KiPs are usually collaborative processes. Thereby, specifying as well as assisting the interactions of multiple knowledge workers in the process enactment is a crucial aspect that should not be overlooked. Both for Declare and MailOfMine, the major efforts have been put in the development and improvement of the control-flow discovery phase. In most of the cases, the workflow of KiPs is unknown a priori and there is an inadequate specification of the knowledge pertaining these processes. Therefore, the research in the context of KiPs clearly benefits from the attention paid to the mining of declarative models, because it can be considered the first necessary step towards their comprehensive management. The declarative specification is still being extended with the specification of process data. Thereby, only a minimal support to data management is currently provided by Declare, and no support at all by MailOfMine. The limited focus on data-oriented modeling and execution may prevent the declarative approach from fully assisting the management of the complex KiPs' life cycles. Specific challenges thus concern the understanding of the link between the evolution of data and the decisions that are taken accordingly, together with a better definition of the role that knowledge workers interpret in the execution of activities. Furthermore, the need to mine process models out of non-conventional and unstructured source of information is crucial.

Although process flexibility increases significantly with declarative modeling approaches, we recognize that, given the characteristics and requirements imposed by KiPs, it becomes increasingly difficult to support them and express the process knowledge in terms of activity-centric languages. The root cause of many of the limitations of activity-centric approaches (either imperative or declarative-constraint-based) in supporting KiPs is identified in the lack of integration of processes and data [23, 40, 56]. Data-centric, object-aware process and case management approaches have recently emerged to overcome these limitations. In an attempt to achieve a complete integration of processes and data, they emphasize the role of data as first-class citizens in process management. In data-centric methodologies, the data perspective is predominant and captures domain-relevant object types, their attributes, their possible states and life cycles, and their interrelations: altogether, they form a complex data structure or *information model*. Such data model enables the identification and definition of the activities that rely on the object-related information and act on it, producing changes on attribute values, relations and object states. Similarly, the case management paradigm focuses on the case (an insurance claim, a customer purchase request, patient case file, etc.) as primary object of interest, and the progress of the case itself is driven by the availability, values, changes and evolution of data objects and their dependencies. Data-centric process management fosters the integration of the main process perspectives, including data, functions, users and processes. In particular, an integrated modeling of domain data and users, as supported by the PHILharmonicFlows framework, has a major impact on process enactment and support. Knowledge workers are explicitly linked to domain-relevant data and the definition of authorization constraints allows supporting multi-user interactions over potentially complex data structures. When data is the main driver for process progression, user involvement explicitly contributes to the overall progression. On the one side, guidance is provided for controlling the interaction with data elements. On the other side, user decisions and commitments influence activity executions and data evolution. Such an approach has found a natural application, for example, in the healthcare domain, where the limited adoption of process management solution for medical processes is often explained with the inability of PMSs to meet flexibility requirements. In particular, the PHILharmonicFlows framework has been evaluated against the complex requirements of healthcare processes [12].

Similarly, initial research efforts show that artifact-centric approaches represent a promising solution for supporting KiPs and case management practices. The artifact-centric approaches (and the GSM meta-model in particular) as a way for supporting adaptive case management, has aroused an increasing interest. This is reflected in the release by the Object Management Group (OMG) of a first standard beta version of the **Case Management Model and Notation (CMMN 1.0)**.[11] CMMN is a meta-model and notation for modeling and graphically expressing a case. Initially conceived as an extension of BPMN 2.0, it has evolved towards a completely different modeling approach: this is also due to its strong link with the business artifacts framework, as both the graphical notation and the operational semantics of CMMN are directly derived from GSM model presented before [47]. CMMN relies on GSM constructs (guards, stages, milestones and sentries), with the additional possibility to unlink milestones from specific stages, define repetition strategies for stages and tasks, and enable late modeling/planning by introducing discretionary elements to be selected at run-time. Specifically, every case is associated with a case file (or information model), which includes and represents all information required as context and data for managing a case. Information in the case file serves as context for raising events, evaluating expressions and defining input and output parameters of tasks. A case plan model defines all the elements that represent the initial plan of the case (tasks, events, rules, constraints, etc.), and all elements that support the further evolution of the plan through run-time planning by case workers. Case roles can be specified to authorize case workers or teams of case workers to perform human tasks, introduce discretionary items at run-time, and raise user events, which influence the proceeding of the case. Run-time planning is enabled by defining in the initial model planning tables that include discretionary items. They can be selected and added to the case plan at run-time by the case worker, possibly constrained by applicability/eligibility rules evaluated over the information model. In addition, repetition rules can be defined to specify under which conditions tasks, stages and milestones can have repetitions. Although the overall case progress is context- and data-dependent and induced by events, conditions and rules, individual tasks that are defined or planned and executed may be linked to predefined procedural processes (e.g., BPMN specifications). This enables a flexible selection and composition of predefined fragments, strengthening the possibility to combine different modeling styles. Run-time planning, even if based on predefined discretionary items, enables late modeling and process adaptation, and can serve as a basis for process evolution. Although proper support for late modeling and run time planning is highly required for KiPs (along with techniques to understand how run-time changes affect running process instances), our analysis shows that little or no support is currently provided.

[11] `http://www.omg.org/spec/CMMN/`

While the conceptual and theoretical foundations of data- and artifact-centric paradigms are well understood, additional research efforts are needed to define clear design methodologies and support process adaptation and evolution requirements. This also relates to the role of process mining and discovery, as adaptivity and evolution in KiPs are linked to the identification of case patterns and events recorded in case histories. Analytical techniques enable a continuous improvement that allows the modeled elements to be modified, extended and potentially introduced into the run-time environment. The initial model may thus evolve over time as the case progresses and as a result of analysis and mining activities performed over the history of closed cases.

8 Conclusion

In this work, we provide a precise characterization of KiPs and, starting from three real-world application scenarios, we devise some general requirements for supporting the life cycle of a KiP. Finally, we present a critical analysis on a number of existing approaches used for supporting KiPs by discussing their efficacy against the devised requirements. Furthermore, we show some recent research techniques that may complement or extend the existing state of the art to this end.

The characteristics and requirements of KiPs force to reconsider the classical process life cycle based on the design–execute&monitor–analyze–re-design sequential steps. The boundary between process design and execution gradually disappears, replaced by a continuous interleaving and overlapping between design, execution and adaptation activities. Although it is possible to foresee the use of templates and fragments as collections of predefined elements to be composed at run-time, in an extreme case the process is completely built from scratch while it is executed, or it has to be discovered by analyzing existing work practices.

Initial research efforts show that data-centric approaches represent a promising solution for supporting KiPs and case management practices. Although object-aware approaches and artifact-centric models at the heart of the CMMN standard can open the way for a new generation of flexible and adaptive case management systems, the level of maturity of existing prototypical frameworks is low if compared to consolidated PMSs. Consequently, the role of these emerging approaches, as well as the potential impact of the upcoming CMMN standard, clearly need further investigation to evaluate the related tools and methods in concrete settings. The advantages with respect to other consolidated approaches have to be verified as well.

Acknowledgements This work has been partly supported by Sapienza through the grants TESTMED, SUPER, "Premio Ricercatori Under-40" and "Assegni di Ricerca 2012". The work of Claudio Di Ciccio has been partly performed when research fellow with InfoSapienza center. With Wirtschaftsuniversität Wien, his research has received funding from the EU Seventh Framework Programme (FP7) under grant agreement 318275 (GET Service).

References

1. van der Aalst WMP (1998) The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems, and Computers 8(1):21–66
2. van der Aalst WMP (2013) Business Process Management: A Comprehensive Survey. ISRN Soft Eng
3. van der Aalst WMP, ter Hofstede AHM, Kiepuszewski B, Barros AP (2003) Workflow Patterns. Distributed and Parallel Databases 14(1):5–51
4. van der Aalst WMP, Adams M, ter Hofstede AHM, Pesic M, Schonenberg H (2009) Flexibility as a Service. In: Database Syst. for Adv. App., DASFAA'09
5. van der Aalst WMP, van Dongen BF, Günther CW, Rozinat A, Verbeek E, Weijters T (2009) ProM: The Process Mining Toolkit. In: BPM (Demos)
6. van der Aalst WMP, Pesic M, Schonenberg H (2009) Declarative workflows: Balancing between flexibility and support. Comp Sc - R&D 23(2):99–113
7. Bahrs J, Müller C (2005) Modelling and analysis of knowledge intensive business processes. In: Proc. of the 3rd Biennial Conf. on Prof. Know. Management, WM'05
8. Bottrighi A, Chesani F, Mello P, Montali M, Montani S, Terenziani P (2011) Conformance Checking of Executed Clinical Guidelines in Presence of Basic Medical Knowledge. In: BPM Workshops
9. BPTrends (2009) Case Management: Combining Knowledge With Process. BPTrends, www.bptrends.com
10. Catarci T, Dix AJ, Katifori A, Lepouras G, Poggi A (2007) Task-Centred Information Management. In: Proc. of the 1st Int. Conf. on Digital Libraries, DELOS'07
11. Catarci T, de Leoni M, Marrella A, Mecella M, Russo A, Steinmann R, Bortenschlager M (2011) WORKPAD: Process Management and Geo-Collaboration Help Disaster Response. IJISCRAM 3(1)
12. Chiao CM, Künzle V, Reichert M (2012) Towards Object-aware Process Support in Healthcare Information Systems. In: Proc. of the 4th Int. Conf. on eHealth, Telemedicine, and Soc. Medicine, eTELEMED'12
13. Cohn D, Hull R (2009) Business Artifacts: A Data-centric Approach to Modeling Business Operations and Processes. IEEE Data Eng Bulletin 32(3):3–9
14. Dalmaris P, Tsui E, Hall B, Smith B (2007) A framework for the improvement of knowledge-intensive business processes. BPM Journal 13(2):279–305
15. Davenport TH (2005) Thinking for a Living: How to Get Better Performance and Results from Knowledge Workers. Harvard Business Rev. Press
16. Davenport TH, Jarvenpaa SL, Beers MC (1996) Improving Knowledge Work Processes. Sloan Management Rev 37
17. De Giacomo G, Lespérance Y, Levesque H, Sardina S (2009) IndiGolog: A High-Level Programming Language for Embedded Reasoning Agents. In: Multi-Agent Programming, Springer US
18. Di Ciccio C, Mecella M (2013) A Two-Step Fast Algorithm for the Automated Discovery of Declarative Workflows. In: CIDM'13, pp 135–142
19. Di Ciccio C, Mecella M (2013) Mining Artful Processes from Knowledge Workers' Emails. IEEE Internet Computing 17(5):10–20
20. Di Ciccio C, Catarci T, Mecella M (2011) Representing and Visualizing Mined Artful Processes in MailOfMine. In: USAB'11, pp 83–94
21. Di Ciccio C, Marrella A, Russo A (2012) Knowledge-intensive Processes: An Overview of Contemporary Approaches. In: KiBP'12
22. Di Ciccio C, Mecella M, Scannapieco M, Zardetto D, Catarci T (2012) MailOfMine – Analyzing Mail Messages for Mining Artful Collaborative Processes. In: Data-Driven Process Discovery and Analysis, Springer, pp 55–81
23. Dumas M (2011) On the Convergence of Data and Process Engineering. In: Proc. of the 15th Int. Conf. on Advances in Databases and Inf. Syst., ADBIS'11
24. Dumas M, van der Aalst WMP, ter Hofstede AHM (2005) Process-aware Information Systems: Bridging People and Software through Process Technology. Wiley-Interscience
25. Dumas M, La Rosa M, Mendling J, Reijers HA (2013) Fundamentals of Business Process Management. Springer
26. Dupras D, Bluhm J, Felty C, Hansen C, Johnson T, Lim K, Maddali S, Marshall P, Messner P, Skeik N (2013) Venous Thromboembolism Diagnosis and Treatment. Institute for Clinical Syst. Improv.
27. Eppler MJ, Seifried P, Röpnack A (2008) Improving Knowledge Intensive Processes through an Enterprise Knowledge Medium. In: Kommunikationsmanagement im Wandel, pp 371–389

28. Gronau N, Weber E (2004) Management of Knowledge Intensive Business Processes. In: 2nd Int. Conf. on Business Process Management, BPM'04
29. Gronau N, Muller C, Uslar M (2004) The KMDL Knowledge Management Approach: Integrating Knowledge Conversions and Business Process Modeling. In: Practical Aspects of Know. Management
30. Günther C, Rinderle S, Reichert M, van Der Aalst WMP (2006) Change Mining in Adaptive Process Management Systems. In: CoopIS'06
31. Harrison-Broninski K (2010) Human Processes. BPTrends, www.bptrends.com
32. ter Hofstede AHM, van der Aalst WMP, Adams M, Russell N (2009) Modern Business Process Automation: YAWL and its Support Environment. Springer
33. Hollingsworth D (1995) The Workflow Reference Model. Tech. Rep. TC00-1003, Workflow Management Coalition
34. Hull R, Damaggio E, De Masellis R, Fournier F, Gupta M, Heath FT III, Hobson S, Linehan M, Maradugu S, Nigam A, Sukaviriya PN, Vaculin R (2011) Business Artifacts with Guard-Stage-Milestone Lifecycles: Managing Artifact Interactions with Conditions and Events. In: 5th Int. Conf. on Distr. Event-Based System, DEBS'11
35. Innocenti P, Ross S, Maceciuvite E, Wilson T, Ludwig J, Pempe W (2009) Assessing Digital Preservation Frameworks: the approach of the SHAMAN project. In: MEDES'09
36. Isik O, Van den Bergh J, Mertens W (2012) Knowledge Intensive Business Processes: An Exploratory Study. In: 45th Hawaii International Conference on System Science, HICSS '12, pp 3817–3826
37. Kemsley S (2011) The Changing Nature of Work: From Structured to Unstructured, from Controlled to Social. In: BPM'11
38. Künzle V, Reichert M (2009) Towards Object-Aware Process Management Systems: Issues, Challenges, Benefits. In: BPMDS'09
39. Künzle V, Reichert M (2011) PHILharmonicFlows: towards a framework for object-aware process management. Journal of Software Maintenance and Evolution: Research and Practice 23(4):205–244
40. Künzle V, Weber B, Reichert M (2011) Object-aware Business Processes: Fundamental Requirements and their Support in Existing Approaches. Journal of Inf Sys Mod and Des 2(2):19–46
41. La Rosa M, Mendling J (2008) Domain-driven process adaptation in emergency scenarios. In: Business Process Management Workshops, Springer, pp 290–297, DOI 10.1007/978-3-642-00328-8_28
42. La Rosa M, Dumas M, ter Hofstede AHM, Mendling J (2011) Configurable multi-perspective business process models. Inf Syst 36(2):313–340
43. Lenz R, Reichert M (2007) IT support for healthcare processes - Premises, challenges, perspectives. Data & Knowledge Engineering 61(1):39–58
44. Leymann F, Roller D (2000) Production Workflow: Concepts and Techniques. Prentice Hall
45. Maggi FM, Bose RPJC, van der Aalst WMP (2013) A Knowledge-Based Integrated Approach for Discovering and Repairing Declare Maps. In: CAiSE'13
46. Malhotra Y (2005) Integrating knowledge management technologies in organisational business processes: getting real time enterprises to deliver real business performance. Journal of Knowledge Management 9(1):7–28
47. Marin M, Hull R, Vaculin R (2012) Data-centric BPM and the Emerging Case Management Standard: A Short Survey. ACM 2012
48. Marjanovic O (2011) Improving Knowledge-Intensive Health Care Processes beyond Efficiency. In: Int. Conf. on Inf. Syst. (ICIS2011)
49. Marjanovic O, Freeze R (2011) Knowledge Intensive Business Processes: Theoretical Foundations and Research Challenges. In: Proc. of the 44th Hawaii Int. Conf. on Sys. Sc., HICSS '11
50. Marjanovic O, Skaf-Molli H, Molli P, Godart C (2007) Collaborative Practice-Oriented Business Processes – Creating a New Case for Business Process Management and CSCW Synergy. In: CollaborateCom 2007
51. Marrella A, Lespérance Y (2013) Synthesizing a Library of Process Templates through Partial-Order Planning Algorithms. In: BPMDS 2013, Springer
52. Marrella A, Mecella M (2011) Continuous Planning for Solving Business Process Adaptivity. In: BPMDS 2011, Springer
53. Marrella A, Russo A, Mecella M (2012) Planlets: Automatically Recovering Dynamic Processes in YAWL. In: OTM 2012, Springer, pp 268–286
54. Marrella A, Mecella M, Sardina S (2014) SmartPM: An Adaptive Process Management System through Situation Calculus, IndiGolog, and Classical Planning. In: KR'14
55. Mendling J, Reijers HA, van der Aalst WMP (2010) Seven process modeling guidelines (7PMG). Information & Software Technology 52(2):127–136
56. Meyer A, Smirnov S, Weske M (2011) Data in Business Processes. EMISA Forum 31(3):5–31
57. Montali M, Chesani F, Mello P, Maggi FM (2013) Towards Data-Aware Constraints in Declare. In: SAC, pp 1391–1396

58. Mundbrod N, Kolb J, Reichert M (2013) Towards a System Support of Collaborative Knowledge Work. In: BPM Workshops, LNBIP, vol 132, Springer Berlin Heidelberg, pp 31–42
59. Pesic M, Schonenberg H, van der Aalst WMP (2007) Declare: Full support for loosely-structured processes. In: EDOC, pp 287–300
60. Pnueli A (1977) The Temporal Logic of Programs. In: 18th Annual Symp. on Found. of Soft. Tech. and Theoretical Comp. Sc. (FSTTCS), pp 46–57
61. Reichert M (2011) What BPM Technology Can Do for Healthcare Process Support. In: Proc. of the 13th Conf. on AI in Medicine, AIME'11, pp 2–13
62. Reichert M, Weber B (2012) Enabling Flexibility in Process-Aware Information Systems – Challenges, Methods, Technologies. Springer
63. Reichert M, Rinderle S, Kreher U, Dadam P (2005) Adaptive Process Management with ADEPT2. In: ICDE, pp 1113–1114
64. Reiter R (2001) Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press
65. Rosenfeld A (2011) BPM: Structured vs. Unstructured. BPTrends, www.bptrends.com
66. Russell N, van der Aalst WMP, ter Hofstede AHM, Edmond D (2005) Workflow Resource Patterns: Identification, Representation and Tool Support. In: Advanced Inf. Syst. Eng., Springer, pp 216–232
67. Russell N, ter Hofstede AHM, Edmond D, van der Aalst WMP (2005) Workflow Data Patterns: Identification, Representation and Tool Support. In: Conceptual Modeling–ER 2005, Springer, pp 353–368
68. Russell N, van der Aalst WMP, ter Hofstede AHM (2006) Workflow Exception Patterns. In: Advanced Information Systems Engineering, Springer, pp 288–302
69. Russo A, Mecella M (2013) On the evolution of process-oriented approaches for healthcare workflows. Int J of Business Process Integration and Management 6(3):224–246
70. Schunselaar DMM, Maggi FM, Sidorova N, van der Aalst WMP (2012) Configurable Declare: Designing Customisable Flexible Process Models. In: CoopIS, pp 20–37
71. Swenson KD (2010) Mastering the Unpredictable: How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done. Meghan-Kiffer Press
72. Vaculin R, Hull R, Heath T, Cochran C, Nigam A, Sukaviriya P (2011) Declarative business artifact centric modeling of decision and knowledge intensive business processes. In: 15th IEEE Int Conf on Enterprise Distr. Object Computing (EDOC 2011)
73. Warren P, Kings N, Thurlow I, Davies J, Buerger T, Simperl E, Ruiz C, Gomez-Perez JM, Ermolayev V, Ghani R, Tilly M, Bösser T, Imtiaz A (2009) Improving Knowledge Worker Productivity - the Active Integrated Approach. BT Tech Journal 26(2)
74. Weber B, Reichert M, Rinderle-Ma S (2008) Change Patterns and Change Support Features–Enhancing Flexibility in Process-Aware Information Systems. Data & Know Eng 66(3):438–466
75. Weber B, Reichert M, Wild W, Rinderle-Ma S (2009) Providing Integrated Life Cycle Support in Process-Aware Information Systems. International Journal of Coop Inf Syst 18(1):115–165
76. Weske M (2007) Business Process Management: Concepts, Languages, Architectures. Springer
77. Westergaard M, Slaats T (2013) CPN Tools 4: A Process Modeling Tool Combining Declarative and Imperative Paradigms. In: BPM (Demos)