Expert Systems With Applications 177 (2021) 114857

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Flexible runtime support of business processes under rolling planning horizons

Irene Barba [a,*], Andrés Jiménez-Ramírez [a], Manfred Reichert [b], Carmelo Del Valle [a], Barbara Weber [c]

[a] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, Spain
[b] Institute of Databases and Information Systems, Ulm University, Germany
[c] Institute of Computer Science, University of St. Gallen, Switzerland

ARTICLE INFO

Keywords:
Process flexibility
Rolling planning horizon
Declarative process
Healthcare process

ABSTRACT

This work has been motivated by the needs we discovered when analyzing real-world processes from the healthcare domain that have revealed high flexibility demands and complex temporal constraints. When trying to model these processes with existing languages, we learned that none of the latter was able to fully address these needs. This motivated us to design TConDec-R, a declarative process modeling language enabling the specification of complex temporal constraints. Enacting business processes based on declarative process models, however, introduces a high complexity due to the required optimization of objective functions, the handling of various temporal constraints, the concurrent execution of multiple process instances, the management of cross-instance constraints, and complex resource allocations. Consequently, advanced user support through optimized schedules is required when executing the instances of such models. In previous work, we suggested a method for generating an optimized enactment plan for a given set of process instances created from a TConDec-R model. However, this approach was not applicable to scenarios with uncertain demands in which the enactment of newly created process instances starts continuously over time, as in the considered healthcare scenarios. Here, the process instances to be planned within a specific timeframe cannot be considered in isolation from the ones planned for future timeframes. To be able to support such scenarios, this article significantly extends our previous work by generating optimized enactment plans under a rolling planning horizon. We evaluate the approach by applying it to a particularly challenging healthcare process scenario, i.e., the diagnostic procedures required for treating patients with ovarian carcinoma in a Woman Hospital. The application of the approach to this sophisticated scenario allows avoiding constraint violations and effectively managing shared resources, which contributes to reduce the length of patient stays in the hospital.

1. Introduction

For more than a decade, there has been an increasing interest in aligning information systems in a process-oriented way (Reichert and Weber, 2012; Weske, 2019). A business process (BP) consists of a set of activities, which jointly realize a specific business goal and whose execution needs to be coordinated in an organizational and/or technical environment (Weske, 2019). Usually, a BP faces numerous constraints to be obeyed during its enactment.

To provide operational support in contemporary process-aware information systems, a business process is usually modeled in an imperative way, i.e., by defining a schema that provides information on how a given set of activities shall be performed. However, imperative approaches to business process management are often too rigid to meet real-world flexibility requirements. As an alternative, providing inherent flexibility, declarative process models have been suggested (van der Aalst et al., 2009; Reichert and Weber, 2012; Debois and Hildebrandt, 2017; Montali, 2010).

1.1. Motivation

The current work has been motivated by the needs we discovered

* Corresponding author.
E-mail addresses: irenebr@us.es (I. Barba), ajramirez@us.es (A. Jiménez-Ramírez), manfred.reichert@uni-ulm.de (M. Reichert), carmelo@us.es (C. Del Valle), barbara.weber@unisg.ch (B. Weber).

https://doi.org/10.1016/j.eswa.2021.114857
Received 9 August 2020; Received in revised form 5 January 2021; Accepted 3 March 2021
Available online 16 March 2021

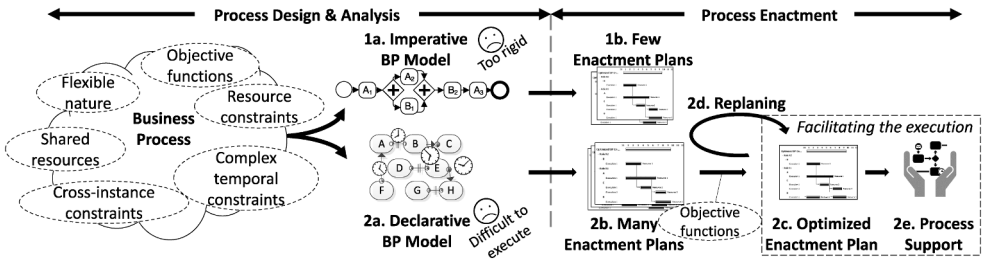

Fig. 1. Overview of our previous related work.

when analyzing sophisticated real-world processes from the healthcare domain, e.g., the diagnostic and therapeutic procedures required in the context of healing patient ovarian carcinoma in a Woman Hospital (Schultheiß et al., 1996; Ovarian cancer (CG122), 2011). In these procedures multiple units (e.g., wards, outpatient department, surgery, labs, radiology, etc.) are involved. When trying to model these processes with existing languages, we learned that none of the latter was able to fully address the needs of patient treatment processes, as detailed in our previous work (Jiménez-Ramírez et al., 2018). To be more precise, we could not find any modelling language that was able to properly deal with both the flexibility demands and the complex temporal constraints set out by considered healthcare processes. This motivated us to design TConDec-R (Barba et al., 2012; Barba et al., 2016), a declarative process modeling language enabling the specification of complex temporal constraints.

On one hand, a declarative process model offers a high degree of execution flexibility to users. On the other, executing a declarative model entails larger efforts for users compared to imperative models (Reichert and Weber, 2012; Schonenberg et al., 2008). While rather few decisions have to be made when executing an imperative model (cf. Steps 1a + b in Fig. 1), for a declarative process model, users may choose among numerous ways of executing it (cf. Steps 2a + b in Fig. 1). Consequently, the decision on how to execute a declarative model is more complex (Haisjackl et al., 2016; Zugal et al., 2015). This complexity further increases for business processes that (1) comprise temporal or cross-instance constraints, (2) require an efficient management of shared resources, and (3) need to optimize objective functions in an uncertain evolving environment (cf. *Business Process* in Fig. 1), as in the scenarios considered. As this might result in sub-optimal enactment plans, advanced user support (cf. Step 2e in Fig. 1) is indispensable when executing declarative models. To enable such runtime support, in previous work we proposed a constraint-based approach for automatically generating optimized enactment plans (cf. Step 2c in Fig. 1) from TConDec-R process models (Barba et al., 2012; Jiménez-Ramírez et al., 2018). Moreover, as explained in Barba et al. (2013), during process enactment this approach allows flexibly adapting the plans if required, considering run-time information as well (cf. Step 2d in Fig. 1).

Many processes can be managed considering our previously proposed approach (e.g., Jiménez-Ramírez et al., 2018). However, it revealed several limitations when applying it to certain real-world scenarios. To be more precise, in many real-world scenarios, like the ones we investigated in the healthcare domain, the enactment of process instances starts continuously over time and the planning horizon can therefore be considered as infinite as there always exist running instances. Generating optimized enactment plans in such scenarios is both challenging and highly complicated as the information required for generating the plans might be unknown, making it necessary to rely on forecasts when making planning decisions (Xie et al., 2003). Note that planning decisions related to later periods need to be generated based on data that might change later (forecasted information) (Tiacci and Saetta, 2012).

1.2. Contribution

To enable the support of scenarios that can be modelled with TConDec-R and for which the enactment of process instances may start continuously over time, we propose the generation of *optimized enactment plans* on a rolling horizon basis with a finite planning horizon (Karimi et al., 2003; Tiacci and Saetta, 2012) (cf. Fig. 2). At a specific point in time T, there exists a set of running process instances. Moreover, there are instances that will potentially start after T and, hence, need to be considered when generating the optimized enactment plans at T taking the planning horizon length (PHL for short) into account. For both instance sets, required future information might be already known (e.g., activities being mandatory for any process instance) and, in addition, unknown future information might exist. Regarding unknown information, forecasts are provided based on previous process instances (Zhao et al., 1995; Tiacci and Saetta, 2012).

When generating an enactment plan for a given planning horizon, at planning point T, both known and forecasted information are considered. The plan is then generated by optimizing the enactment of all instances (cf. sets *a* and *b* in Fig. 2) according to a given objective function and fulfilling the specification of the process behavior (i.e., the declarative BP model). Although a plan for the complete planning horizon is created, usually, only certain decisions are carried out short term, while others are deferred to future planning points. The time spent between two successive planning points is determined by the replanning period (i.e., RP). This way, at planning point T + RP, the planning horizon is rolled forward and the same procedure is repeated considering the

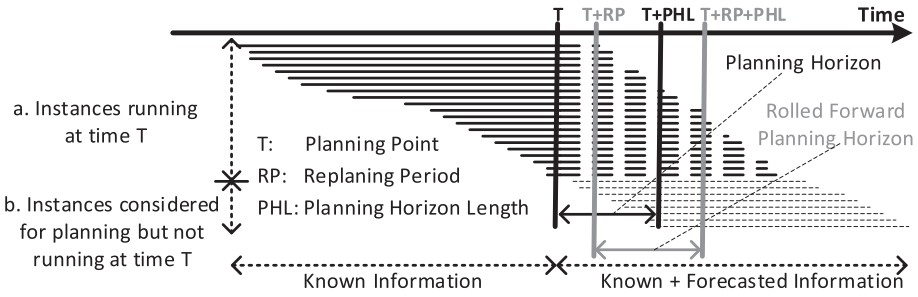

Fig. 2. Replanning under rolling planning horizons.

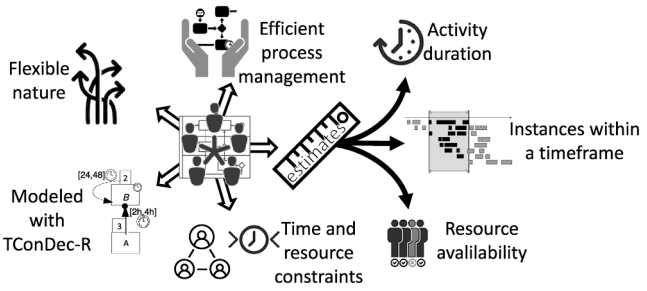

Fig. 3. Scenarios in which the proposed approach can be successfully applied.

updated information in the event log, the information that is known, and the forecasts. Thereby, decisions of the planning horizon related to T, which were not taken before T + RP, are free to be replanned at T + RP.

To validate the approach, we apply it to a particularly challenging scenario from the investigated ones, i.e., the diagnostic and therapeutic procedures required for treating patients with ovarian carcinoma in a Woman Hospital (Schultheiß et al., 1996; Ovarian cancer (CG122), 2011). Note that the approach is not restricted to healthcare, but can be applied in other domains (e.g., engineering processes that imply high demands regarding flexibility and temporal constraint management) as well. To be more precise, the proposed approach targets at scenarios characterized by (1) their flexible nature, (2) cross-process constraints involving the time and resource perspectives, and (3) the need to efficiently manage shared resources in accordance with a given optimization criterion. Note that our approach can be only applied to scenarios that allow performing estimates on activity durations, resource availability, and the number of process instances started within a specific timeframe. Moreover, it must be possible to properly model the respective processes with TConDec-R (cf. Fig. 3).

This article enhances our previous work by addressing a fundamental primary research question (PRQ) not considered so far:

(PRQ) Is the proposed approach useful for coping with real process scenarios in which the enactment of process instances starts continuously over time?

With the goal of answering PRQ, this paper significantly extends and improves our previous work by (1) enabling *run-time flexibility under rolling planning horizons*, (2) improving the applicability of the approach to real-world scenarios, and (3) performing an empirical evaluation along a particularly challenging healthcare process scenario. On one hand, these extensions significantly broaden the applicability of the proposed approach; on the other, they show that the approach can be successfully applied to complex scenarios that present characteristics similar to the ones we observed in healthcare scenarios.

The remainder of this paper is organized as follows: Section 2 discusses related work, while Section 3 gives background on our previous works. Section 4 formalizes the addressed problem. Section 5 summarizes the original contribution of the paper, i.e., it shows how the optimized enactment plans are generated under rolling planning horizons. Section 6 then discusses how the generated optimized enactment plans can be used for improving process support. Section 7 describes the performed evaluation that is based on the application of the proposed approach to the real-world scenario from healthcare. Section 8 discusses the current state of our research, including its limitations, and, finally, Section 9 concludes the paper.

Table 2
Selected process time patterns (adopted from Lanz et al. (2014), Lanz et al. (2016)).

Time pattern (TP)	Example
TP1 (*Time Lags between two Activities*) enables the definition of different kinds of time lags between two activities.	*The time lag between subscribing to a Master thesis project and submitting the thesis must not exceed 6 months.*
TP2 (*Durations*) allows specifying the duration of process activities.	*Processing 100 requests must not take longer than 1 s.*
TP5 (*Schedule Restricted Element*) allows restricting the enactment of a particular activity by a schedule.	*In a hospital, full lab tests may solely be ordered from Monday to Friday between 8 am and 5 pm.*
TP6 (*Time-based Restrictions*) restricts the number of times a specific process element (e.g., activities or instances) may be executed within a given timeframe.	*For a specific lab test, at least 5 different blood samples have to be taken from the patient within 24 h.*

2. Related Work

This section presents related work on (1) declarative process modeling languages that support temporal constraints, (2) scheduling, (3) decision support systems, and (4) process enactment. The main findings are summarized in Table 1.

There exist several proposals for managing temporal constraints in various domains. In Jiménez-Ramírez et al. (2018), we reviewed declarative process modeling languages that support the temporal constraints necessary for modeling a sophisticated real-world process scenario from healthcare trough the analysis of supported *time patterns*. The latter are solutions for representing commonly occurring temporal constraints in Process-Aware Information Systems (PAISs), cf. Table 2 (Lanz et al., 2014; Lanz et al., 2016). To be more precise, several works (Montali et al., 2013; Hildebrandt et al., 2013; Maggi and Westergaard, 2014; Maggi, 2014; Zeising et al., 2014; Jiang et al., 2016; Burattin et al., 2016; Schönig et al., 2016; Barba et al., 2012; Mans et al., 2010; Borrego et al., 2020; Käppel et al., 2019; Mertens et al., 2017; Xu et al., 2020) were identified as relevant in the context of our research. In particular, the current work considers 4 of the 10 time patterns suggested in Lanz et al. (2014) as they refer to aspects directly related to the considered healthcare processes (Jiménez-Ramírez et al., 2018). We concluded that (1) time patterns TP1 and TP2 (cf. Table 2) are well supported, (2) time pattern TP5 (cf. Table 2) is only supported by Mans et al. (2010), Barba et al. (2012), Mertens et al. (2017) and Borrego et al.

Table 1
Comparison with related work.

Topic	Main contributions of the current work
Temporal declarative process modeling languages (Montali et al., 2013; Hildebrandt et al., 2013; Maggi and Westergaard, 2014; Maggi, 2014; Zeising et al., 2014; Jiang et al., 2016; Burattin et al., 2016; Schönig et al., 2016; Mans et al., 2010; Borrego et al., 2020; Käppel et al., 2019; Slaats et al., 2013; Mertens et al., 2017; Xu et al., 2020)	The proposed approach allows for time-based constraints, restricting the number of times a particular process element may be executed within a pre-specified timeframe.
Scheduling (Hoenisch et al., 2013; Berbner et al., 2007; Oh et al., 2011; Masdari et al., 2016; Karimi et al., 2003; Xie et al., 2003; De Araujo et al., 2007; Tiacci and Saetta, 2012)	The proposed approach allows allocating resources at regular intervals, taking the current process context into account. The proposed approach supports complex temporal constraints and cross-instance coordination.
Decision support systems (Lanz et al., 2013; Eder et al., 2003; Eder and Pichler, 2005; van Dongen et al., 2008; van der Aalst et al., 2011; Pesic et al., 2007)	The proposed approach considers the optimization of given objective functions.
Enactment of declarative process models (Carvalho et al., 2013; Goedertier et al., 2008; Schönig et al., 2018; Marquard et al., 2015; Ackermann et al., 2018; Montali et al., 2013; Pesic, 2008; Westergaard and Maggi, 2012; Mertens et al., 2019)	The proposed approach allows for an efficient replanning during process enactment while guiding users based on the optimization of a given objective function.

(2020), and (3) TConDec-R (Barba et al., 2012) is the only approach supporting time pattern TP6 (cf. Table 2).[1] To the best of our knowledge, therefore, only TConDec-R supports the modeling of business processes enhanced with the temporal constraints related to the time patterns.

While existing scheduling proposals (e.g., Hoenisch et al., 2013; Berbner et al., 2007; Oh et al., 2011; Masdari et al., 2016) mainly consider partial information for assigning resources to activities at run-time (i.e., *dynamic scheduling*), we suggest allocating resources at regular intervals, taking the current process context into account as well. Note that this allows us to improve overall process enactment as global information can be analyzed. Moreover, there exist proposals related to scheduling in *lot sizing* scenarios with uncertain demand (e.g., Karimi et al., 2003; Xie et al., 2003; De Araujo et al., 2007; Tiacci and Saetta, 2012). Unlike our approach, these works neither consider complex temporal constraints nor cross-instance process coordination, ensuring constraints across different process instances.

Regarding decision support systems, Lanz et al. (2013) presents an approach for enabling the proper visualization of personal schedules from well-structured process models with temporal constraints. Unlike our approach, Lanz et al. (2013) starts from an imperative model in such a way that the generated schedules just reflect the source imperative model. Moreover, Lanz et al. (2013) does not consider the optimization of any objective function. Similarly, Eder et al. (2003) presents an approach for providing actors with personalized schedules, which is based on a probabilistic time-aware workflow system that uses duration histograms. Unlike our approach, Eder et al. (2003) uses the personal schedules only for workload prediction (e.g., detecting future bottlenecks and upcoming violations of time constraints as early as possible), but not for providing directives to the process participants (Eder et al., 2003). In a related way, a probabilistic workflow system for time prediction is presented in Eder and Pichler (2005). Similarly, van Dongen et al. (2008) proposes a service that predicts the completion time of process instances using non-parametric regression. However, the focus of Eder and Pichler (2005) is more on scheduling and escalation; unlike our approach, Eder and Pichler (2005) assumes that the workflow is known beforehand and is stable (van der Aalst et al., 2011). Moreover, both Eder et al. (2003) and Eder and Pichler (2005) provide design-time support, whereas our approach enables run-time support as well. Pesic et al. (2007) provides run-time support for dynamic changes in the context of constraint-based business process models. Unlike the proposed approach, Pesic et al. (2007) neither directly supports the selected time patterns (cf. Table 2) nor optimization of objective functions.

Optimizing a given objective function fosters decision support when executing declarative process models. Note that this complements other decision support approaches that utilize the data perspective (Montali, 2010; Montali et al., 2013; Maggi et al., 2013; Borrego and Barba, 2014; Mertens et al., 2017; Slaats et al., 2013).

Additionally, there exist proposals that foster the enactment of declarative process models. For the Declare family of languages (to which TConDec-R belongs), for example, Pesic (2008) proposed the generation of a non-deterministic finite state automaton from declarative specifications based on linear temporal logic (LTL), which exactly represents those traces satisfying the LTL formulas. Later, this approach was extended to include the time (Westergaard and Maggi, 2012) and data (Montali et al., 2013) perspectives as well. As a drawback, the automatic generation of state automaton from declarative specifications is NP-complete and, unlike to our approach, no heuristics have been used. Similarly, CLIMB generates quality traces from declarative specifications (Montali, 2010). Then, the best traces are selected for enactment. Unlike our approach, CLIMB does not consider the resource perspective. Its features were integrated into MP-Declare whose enactment is supported by Ackermann et al. (2018) through the use of Alloy and its satisfiability solver. In Carvalho et al. (2013), the graph-based approach ReFlex is proposed to improve the enactment capabilities of Declare. All these approaches neither provide an efficient replanning feature nor do they support the optimization of a given objective function when guiding users during process enactment.

There exist other declarative approaches that support process enactment as well. The EM-BrA^{2}CE textual framework (Goedertier et al., 2008) extends the semantics of business vocabularies and business rules for declarative process models, enabling their enactment in a service-oriented architecture. Similarly, DPIL Navigator (Schönig et al., 2018) enables the enactment of textual DPIL models, including the support of certain workflow resource and data patterns. A more human-understandable approach is presented by Marquard et al. (2015), which proposes a process engine for DCR Graphs. This engine supports the enactment of declarative models in a collaborative environment. Although the engine can be used to generate traces, it lacks a more extensive support for the data and resource perspectives. More recently, Mertens et al. (2019) introduced the generic declarative process engine DeciClareEngine-tool, which does not enforce the use of any particular declarative language. This engine supports data, resource and deadline constraints. Unlike our proposal, however, it lacks support for more complex time constraints.

In a nutshell, the primary aim of all these approaches resides on calculating the possible next actions, while satisfying the constraints of the declarative specification. Although this article does not elaborate deeply on this specific feature, unlike these related approaches, it provides advanced features like replanning (during process enactment), optimizing an objective function, and supporting sophisticated time and cross-instance constraints that, to the best of our knowledge, have not been considered by existing approaches for declarative process enactment.

3. Backgrounds

This section presents backgrounds needed for understanding this work. Section 3.1 presents the declarative process modeling language TConDec-R (Barba et al., 2012), whereas Section 3.2 shows how optimized enactment plans can be obtained from TConDec-R models (Jiménez-Ramírez et al., 2018). Finally, Section 3.3 introduces a sophisticated real-world process scenario from healthcare and shows how it can be modeled with TConDec-R (Jiménez-Ramírez et al., 2018).

3.1. The TConDec-R language

Fundamental to TConDec-R is Declare (Pesic, 2008), i.e., a declarative modeling language that allows specifying a set of activities together with the constraints to be obeyed during process enactment such that business goal achievement can be ensured. We use TConDec-R for specifying activities, their behavioral constraints and temporal constraints (with the latter allowing for the coverage of fundamental time patterns, cf. Table 2). TConDec-R process models are denoted as *constraint-based* as they comprise information about (1) the activities that may be performed during process enactment and (2) the constraints to be obeyed in this context. Note that TConDec-R is a constraint-based language and, hence, enables loosely specified process models (van der Aalst, 2009) allowing users to defer modeling decisions to the run-time (Reichert and Weber, 2012).

Definition 1. (**TConDec-R activity**) A **TConDec-R activity** $act = (a, dur, role)$ refers to a process activity a with its estimated duration dur and the role of the required resource.

Definition 2. (**TConDec-R process model**) A **TConDec-R process model** $TCRM = (Acts, C_T, Res)$ is an extended constraint-based process model, where $Acts$ corresponds to a set of TConDec-R activities, C_T corresponds to a set of constraints, which may include any control-flow

[1] Note that some of these proposals (e.g., Mertens et al., 2019; Borrego et al., 2020) partially address TP6. In general, the scope of TP6 includes both *activity* and *process instance*, whereas these proposals only consider the activity scope.

constraint supported by Declare as well as any temporal constraint related to the time patterns from Table 2, and *Res* corresponds to the resources available for process execution.

TConDec-R constraints are specified according to the graphical notation introduced by Declare (Pesic, 2008) and the one proposed in Lanz et al. (2016) for visualizing temporal constraints.[2] For creating TConDec-R models, a web-based tool was implemented (Jiménez-Ramírez et al., 2013; Barba et al., 2012).[3]

A *process instance* (cf. Definition 3) represents the concrete execution of a TConDec-R process model (i.e., a business case) and its execution state reflected by the *execution trace* (cf. Definition 4).

Definition 3. (**Process instance**) Let *TCRM= (Acts, C_T, Res)* be a TConDec-R process model. Then: A **process instance** $PI^{id} = (TCRM, \sigma^{id})$ on *TCRM* has a unique identifier *id* and is defined by *TCRM* as well as a corresponding trace σ^{id} (cf. Def. 4).

Definition 4. (**Trace**) Let *TCRM= (Acts, C_T, Res)* be a TConDec-R process model. Then: A **trace** σ^{id} is composed of a sequence of (1) start and completion events related to activity executions $a_i^{id}, a \in Acts$ of process instance *id*, and (2) events related to a resource becoming available or unavailable, i.e., events can be

- $start(a_i^{id}, R_{jk}, T)$, i.e., the i-th execution of activity *a* in the context of process instance *id*, using the k-th resource with role *j*, was started at time *T*.
- $comp(a_i^{id}, T)$, i.e., the i-th execution of activity *a* in the context of process instance *id* was completed at time *T*.
- $available(R_{jk}, T)$, i.e., the k-th resource with role *j* became available at time *T*.
- $unavailable(R_{jk}, T)$, i.e., the k-th resource with role *j* became unavailable at time *T*.

When proceeding with the execution of a TConDec-R process model, information regarding the executed activities is recorded in an event log (cf. Definition 5). In general, an event log comprises a set of execution traces.

Definition 5. (**Event Log**) An **event log** is composed of the traces related to the execution of a set of process instances.

3.2. From TConDec-R models to optimized enactment plans

To generate optimized enactment plans for a specific TConDec-R process model (i.e., plans that consider the optimization of the given objective function), we proposed a constraint-based approach in previous work (Jiménez-Ramírez et al., 2018). This approach utilizes the constraint programming (CP) paradigm for modelling and solving planning and scheduling problems in an effective way (Rossi et al., 2006). As core idea of the CP paradigm, the user defines the problem as a set of decision variables and a set of constraints between these variables and the input data. Then, a constraint-solver is used to find the values of the decision variables meeting the constraints and, when required, leading to the optimization of the given objective function. To be more precise, the goal of the proposed approach is to determine a plan (i.e., start times of process activities) taking into account (1) all process constraints, (2) the shared resources, and (3) an optimization criterion. Since the generation of optimal plans has NP-complexity (Garey and Johnson, 1979), it is not possible to ensure the optimality of the generated plans for all cases.

To solve a problem through constraint programming, it needs to be modelled as a *constraint satisfaction problem* (CSP, cf. Def. 6).

Definition 6. (**Constraint Satisfaction Problem**) A **CSP** $P = (V, D, C_{CSP})$ is composed of a set of variables *V*, a set of domains of values *D* for all variables, and a set of constraints C_{CSP} between variables, such that each constraint (1) represents a relation between a subset of variables and (2) specifies the allowed combinations of values for these variables.

A solution to a CSP consists of assigning values to CSP variables, such that the assignments satisfy all constraints. Similar to CSPs, constraint optimization problems (COPs, cf. Def. 8) require solutions that optimize certain objective functions (cf. Def. 7).

Definition 7. (**Objective Function**) An **objective function** *OF* is a function whose maximum or minimum shall be determined.

Example 1. (**Objective function**) The objective function to be optimized in the considered healthcare scenario is defined as minimizing the length of patient hospital stays.

Definition 8. (**Constraint Optimization Problem**) A **COP** $P_{OF} = (V, D, C_{CSP}, OF)$ is a CSP $P = (V, D, C_{CSP})$ including the objective function *OF* to be optimized.

In the proposed constraint-based approach, each TConDec-R activity (cf. Def. 1) is modeled as a *repeating activity* (cf. Def. 9).

Definition 9. (**Repeating and scheduling activities**) Let $TCRM = (Acts, C_T, Res)$ be a TConDec-R process model and $PI^{id} = (TCRM, \sigma^{id})$ be a related process instance. Then: A **repeating activity** $ra^{id} = (a, dur, role, nt, sacts)$ related to PI^{id} corresponds to a TConDec-R activity $act = (a, dur, role)$ that may be executed several times during instance enactment. Such repeating activity is described in terms of the estimated duration of the process activity (dur), the role of the resource required for activity enactment (role), and a CSP variable *nt* specifying the number of times the process activity may be executed (i.e., the number of scheduling activities related to ra^{id})[4]. Moreover, *sacts* corresponds to the sequence of scheduling activities related to this repeating activity. Thereby, a **scheduling activity** $a_i^{id} = (ra^{id}, st, et, sel, res)$ corresponds to the i-th enactment of repeating activity ra^{id} with *st*/*et* constituting CSP variables that indicate the start/end time of activity enactment. Moreover, *sel* corresponds to a binary CSP variable that indicates whether such activity is selected for execution[5]. Moreover, *res* corresponds to a CSP variable representing the resource used for activity enactment.

Based on these definitions, TConDec-R models are represented as constraint optimization problems (i.e., COPs), resulting in COP-TConDec-R problems (cf. Def. 10).

Definition 10. (**COP-TConDec-R problem**) Let $TCRM = (Acts, C_T, Res)$ be a TConDec-R process model (cf. Def. 2), $\#inst$ be the number of planned process instances, and *RActs* be the set of repeating activities related to *TCRM* and $\#inst$ (i.e., $RActs = \{ra^{id}, \forall id \in [1, \#inst] | ra^{id} = (a, dur, role, nt, sacts), \text{ with } (a, dur, role) \in Acts\}$). Then: A **COP-TConDec-R problem** related to *TCRM*, $\#inst$, and *RActs* corresponds to a CSP

[2] A complete formalization of the TConDec-R constraints can be obtained via the following URL: https://doi.org/10.5281/zenodo.4387184

[3] Available via http://azarias.lsi.us.es/TCR/ModelChecker.

[4] Note that the value of *nt* (as for any CSP variable) will be determined by the solution to the CSP/COP. As for any CSP variable, lower and upper bounds of the variable domain (i.e., minimum and maximum value, respectively, that can be assigned to *nt* in a valid solution for the CSP) need to be established in the CSP/COP. As detailed in Definition 10, the lower bound for *nt* is always set to 0, whereas the upper bound is set to the maximum cardinality the related process activity may have (denoted as *MAXNT* in Definition 10). Initially, the latter is established through a rough estimate that considers the maximum mandatory cardinality of all process activities (stated by the existence constraints in the constraint-based process model).

[5] Note that whether a scheduling activity a_i^{id} is selected for enactment is directly related to the *nt* variable of the associated repeating activity ra^{id}, i.e., $(1 \leqslant i \leqslant ra^{id}.nt \rightarrow a_i^{id}.sel = 1) \wedge (i > ra^{id}.nt \rightarrow a_i^{id}.sel = 0)$.

$P_{OF} = (V, D, C_{CSP}, OF)$ with

- V being a set that comprises all variables of the CSP model, i.e., $V \equiv \{ra^{id}.nt | ra^{id} \in RActs\} \cup \{a_i^{id}.st, a_i^{id}.et, a_i^{id}.sel, a_i^{id}.res \,|\, a_i^{id} \in ra^{id}.sacts, ra^{id} \in RActs\}$.
- D being a set that covers all value domains of the respective variables from V, i.e., $D \equiv \{D(ra^{id}.nt), D(a_i^{id}.st), D(a_i^{id}.et), D(a_i^{id}.sel), D(a_i^{id}.res) \,|\, a_i^{id} \in ra^{id}.sacts, ra^{id} \in RActs\}$, where
  - $D(ra^{id}.nt) = [0, MAXNT(ra^{id})]$ with $MAXNT(ra^{id})$ being an upper bound for the number of times ra^{id} may be executed,
  - $D(a_i^{id}.st) = [0, MAXOCT]$ with MAXOCT being an upper bound for the overall completion time of the process,
  - $D(a_i^{id}.et) = [0, MAXOCT]$ with MAXOCT being an upper bound for the overall completion time of the process,
  - $D(a_i^{id}.sel) = \{0, 1\}$, and
  - $D(a_i^{id}.res) = [0, |Res(a.role)| - 1]$ with $|Res(a.role)|$ being the number of resources with role $a.role$ in the process model $TCRM$.
- C_{CSP} being a set that comprises the resource constraints as well as the TConDec-R constraints included in C_T (cf. Def. 2). Furthermore, C_{CSP} states that a specific enactment of a repeating activity $ra^{id} \in RActs$ precedes the next enactment of the same activity, i.e., $\forall a_i^{id} \in ra^{id}.sacts : a_i^{id}.et \leqslant a_{i+1}^{id}.st$. Moreover, variable nt is directly related to the variable sel of the associated scheduling activities, i.e., $\forall a_i^{id} \in ra^{id}.sacts : (1 \leqslant i \leqslant ra^{id}.nt \rightarrow a_i^{id}.sel = 1) \wedge (i > ra^{id}.nt \rightarrow a_i^{id}.sel = 0)$ for each repeating activity $ra^{id} \in RActs$.
- OF is the objective function to be optimized.[6]

Based on this COP, we implemented a constraint-based approach that generates an optimized enactment plan (cf. Def. 11) for a given TConDec-R process model, as detailed in our previous work (Jiménez-Ramírez et al., 2018).

Definition 11. (**Enactment plan**) Let $TCRM = (Acts, C_T, Res)$ be a TConDec-R process model (cf. Def. 2), $\#inst$ be the number of process instances planned from TCRM, $RActs$ be the set of repeating activities related to $TCRM$ and $\#inst$, and $P_{OF} = (V, D, C_{CSP}, OF)$ be the COP-TConDec-R problem related to $TCRM$ and $\#inst$. Then: An **enactment plan**

$EP = \{(ra^{id}.nt' \,|\, ra^{id} \in RActs, id \in [1, \#inst]) \cup (a_i^{id}.st', a_i^{id}.et', a_i^{id}.res', a_i^{id}.sel', \text{with } a_i^{id} \in ra^{id}.sacts)\}$ is a specific way to execute a number #inst of process instances of the $TCRM$ model. In the proposed approach, EP is generated by solving P, i.e., by instantiating all CSP variables of P. This way, $ra^{id}.nt'$ is the instantiation of $ra^{id}.nt$ in the solution to P, $a_i^{id}.st'$ is the instantiation of $a_i^{id}.st$ in the solution to P, and so on. Thus, for each planned process instance an enactment plan comprises (1) the number of times each activity is executed, (2) the start and completion times of each activity enactment (i.e., activity instance), and (3) the resource used for each activity enactment.

3.3. Healthcare process scenario

This section details a selected real-world process scenario from healthcare that we consider in the following. Section 3.3.1 introduces the scenario and discusses related challenges, whereas Section 3.3.2 provides further details and the way this scenario can be modeled with TConDec-R.

3.3.1. Scenario context

The selected scenario (Jiménez-Ramírez et al., 2018) deals with the scheduling of surgeries and their preparations in the context of ovarian carcinoma (Schultheiß et al., 1996; Ovarian cancer (CG122), 2011). Patients with ovarian carcinoma are treated in the Women's Hospital (WH). In this context, the available resources of the Women's Hospital (i.e., staff and medical equipment) need to be efficiently and effectively allocated. Currently, the planning and scheduling of the diagnostic investigations is done manually (i.e., by phone calls) by ward staff, resulting in frequent constraint violations in practice. For example, if a required time lag between two medical examinations, Ex1 and Ex2, is not considered at planning time, which happens rather often in clinical practice, this is only detected when trying to perform Ex2. Without obeying the required time lag, however, Ex2 would not yield any meaningful result and, hence, would have to be aborted and rescheduled. As a consequence, Ex2 might be delayed for some time, which, in turn, delays the surgery. Furthermore, patient waiting times are rather high as appointments are not fixed beforehand. Therefore, the patient is taken to the respective department at 8 am and, worst case, then might have to wait the entire morning until her examination starts (at 11 am or later).

Usually, it is not sufficient to schedule examinations in a way satisfying all constraints. Additionally, it must be ensured that surgeries take place as soon as possible after admitting the patient; i.e., the length of patient stays shall be minimized, which should be covered by a corresponding objective function. Note that this consitutes a challenging task when considering the schedules of all patients, often resulting in suboptimal schedules. The latter often lead to unnecessarily long hospital stays and, thus, costs, e.g., due to waiting times caused by bad planning.

Regarding flexibility needs, the planned schedules might become inappropriate during process enactment, e.g., due to inaccurate estimates, process participants not obeying the schedule, or unexpected absence of resources. Moreover, scheduled patient examinations within a given timeframe must not be treated in isolation from other patients as the latter are continuously admitted to the WH. Note that the situation becomes aggravated due to the fact that the Women's Hospital does not exactly know all future information, but has to rely on forecasts instead when planning decisions.

In the considered scenario, preparing of patients for the surgery may also require examinations not carried out in the WH itself, but in external departments, e.g., Radiology department. These examinations need to be considered in the context of scheduling as well. Moreover, when a medical examination is performed at such an external location, a shuttle service needs to be organized for transporting the patient. Currently, the shuttle service for the trip back is requested after finishing the patient's examination, i.e., the patient needs to wait for the shuttle, which takes around 15–20 min. Usually, this delays other examinations as well as the surgery itself.

3.3.2. Scenario details and TConDec-R model for the scenario

On average, two patients with the diagnosis of ovarian carcinoma are admitted to the WH per day. Each process instance corresponds to the set of activities related to the treatment of one particular patient, as described in the following. After admitting a patient to the gynecological ward, one of the two ward physicians visits and examines the patient. Afterwards, the physician orders and schedules a number of medical examinations, which need to be performed before the surgery may take place. Additionally, the patient needs to be examined by an anesthetist who may then request additional medical examinations before the surgery if required. Some of the examinations are not carried out by the WH itself, but are provided by other clinical departments, which may be either internal or external (i.e., placed at a different location) to the WH. In the given scenario, five external departments – Endoscopy Department (ED), Radiology Department (RD), Comprehensive Cancer Centre (CCC), Otolaryngology Department (OD), and Neurology Department (ND) – are involved as well as an internal department with two units, i.e., Ultrasound Unit (UU) and Laparoscopy Unit (LU). Each department has limited resources and provides services to many other departments

[6] As an example consider minimizing the overall completion time of all process instances.

Table 3
Processes relevant in the context of the considered clinical scenario.

ID	Description	Dur	*Unit*/*Dep*	%*Req*
Ex0	First visit and examination of the patient	30 m	WH	100
Ex1	Pelvic Ultra-sound Imaging	30 m	UU	100
Ex2	Cystoscopy & Rectoscopy	2h30m	ED	100
Ex3	Uretero Pyelography	1h30m	RD	100
Ex4	CT scanning	45 m	RD	60
Ex5	Magnetic Resonance Imaging	1h15m	RD	40
Ex6	Colonoscopy	2h15m	CCC	100
Ex7	Colon Contrast Imaging	3h30m	CCC	40
Ex8	X-ray of the gastrointestinal tract	1h15m	RD	35
Ex9	Chest X-ray	30 m	RD	85
Ex10	Blood test	10 m	WH	70
Ex11	Laparoscopy	1 h	LU	100
Ex12	Doppler examinations	30 m	UU	20
Ex13	Medical council with the OD	60 m	OD	20
Ex14	Medical council with the ND	30 m	ND	10
AN	The patient is examined and interviewed by an anesthetist	1 h	WH	100
SU	The surgery is performed	2–6 h	WH	75
		h = hour(s), m = minutes		

and clinics respectively (including WH). In order to ensure a fair use of the shared resources, a particular department may only order a maximum number of a specific examination from the respective provider per day.

All relevant processes and activities of the considered scenario are summarized in Table 3: Column *ID* corresponds to the identifier of the activity, *Dur* to its average duration, and *Unit*/*Dept* to the unit/department the activity is performed at. Finally, %*Req* expresses the frequency with which the respective examination is requested for a particular patient. For example, %*Req* = 100 expresses that the examination is ordered once for every patient.

Concerning the surgery of a particular patient and the preparations required for this surgery, several constraints need to be obeyed. As detailed in Jiménez-Ramírez et al. (2018), the considered scenario can be properly described with TConDec-R resulting in the model shown in Fig. 4 (Jiménez-Ramírez et al., 2018). As can be seen in Fig. 4, each activity is depicted together with its unary constraints (e.g., existence constraint) as well as its properties (i.e., duration and unit or department where it is performed). For example, Ex6 (i.e., colonoscopy) needs to be performed exactly once for each patient and must occur after 8:00 and before 12:00. Additionally, some binary constraints are depicted. For example, Ex6 has a precedence constraint with Ex7 (i.e., colon contrast imaging) that establishes a time lag of at least 6 days between both examinations.

4. Problem formalization

In many real-world scenarios, the requested services (e.g., medical examinations to be performed) may vary over time. Scenarios with uncertain demands are often related to lot sizing problems (Tiacci and Saetta, 2012; Karimi et al., 2003), i.e., the process instances to be planned and scheduled within a specific timeframe must not be considered in isolation from the instances planned for future timeframes. This situation occurs if the enactment of process instances starts continuously over time. In this scenario, the planning horizon length (cf. Def. 13) may be considered as infinite at any planning point (cf. Def. 12) as there always exist running process instances.

Definition 12. (**Planning point**) A **planning point** *T* is defined as a temporal point during the execution of a business process in which an optimized enactment plan is generated, i.e., planning is performed at *T*.

Definition 13. (**Planning horizon length**) Let *T* be a planning point. Then: The **planning horizon length** *PHL* can be defined as the length of the time period to be planned at a specific planning point *T*.

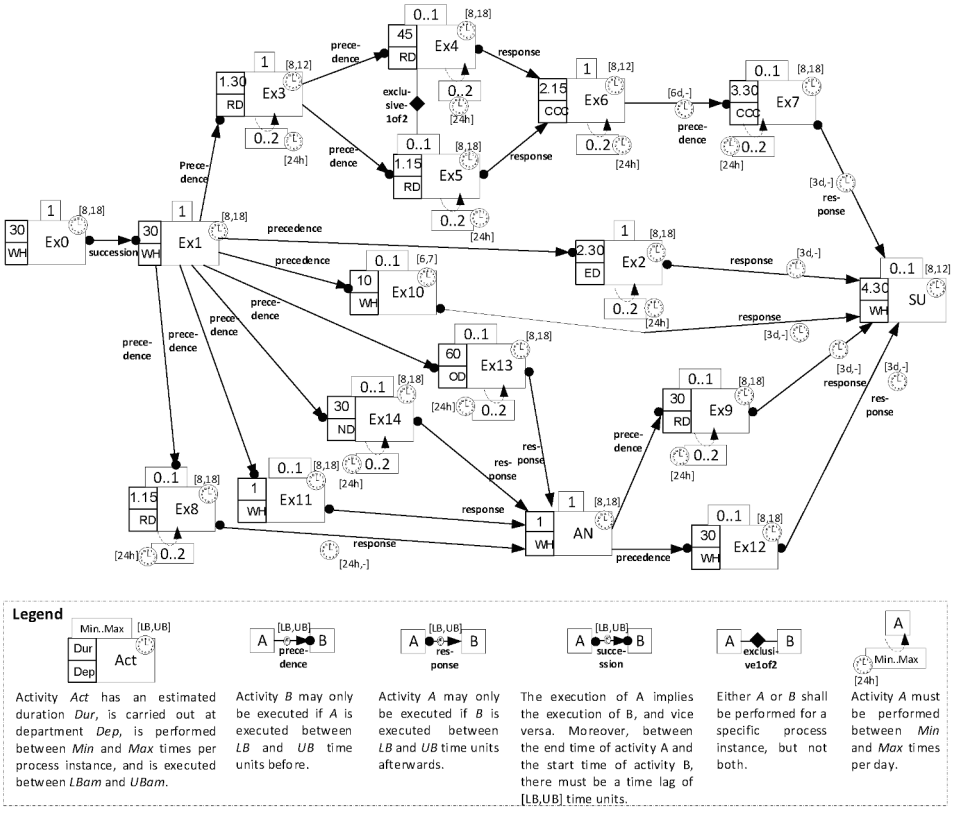

Fig. 4. Simplified TConDec-R model for the scenario.

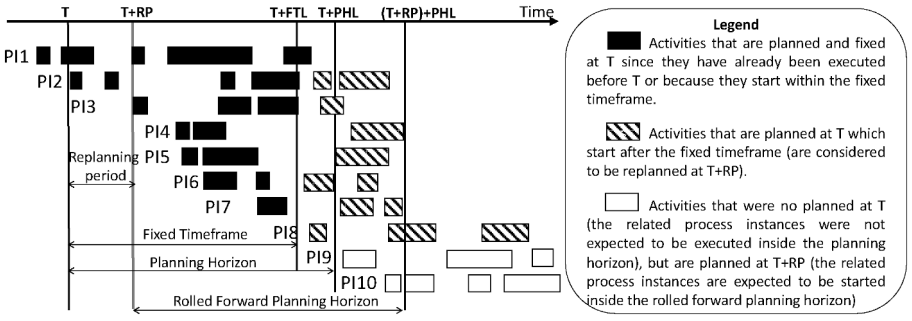

Fig. 5. Planning process instances under a rolling planning horizon.

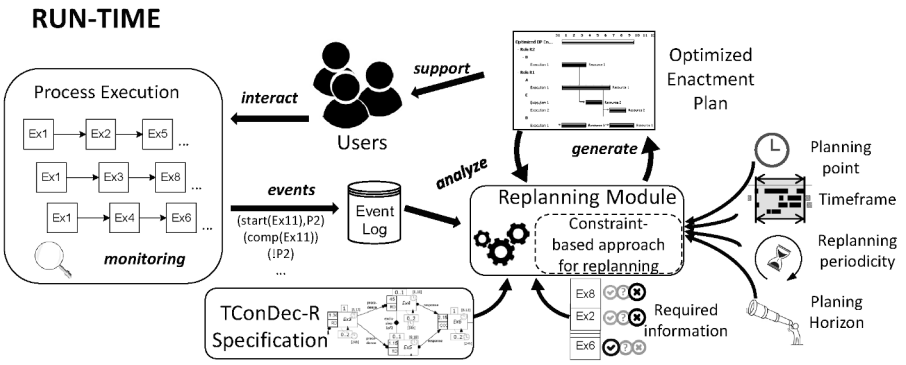

Fig. 6. Planning under rolling planning horizons.

Generally, running process instances differ from each other regarding their respective enactment state, e.g., at time T, a particular instance may have just been started, whereas another one may be almost finished or be expected to start soon. Generating optimized enactment plans in such scenarios is challenging and complicated. While there are confirmed activities regarding already known process instances, the requirement of some activities might be unknown, making it necessary to rely on forecasts when making planning decisions (Xie et al., 2003). To be more precise, there is a set of process instances to be considered at T for performing the planning. Therefore, some information related to these process instances is required (cf. Def. 14). This required information is either known or unknown at T. The unknown information is forecasted based on data that might change during the next periods (Tiacci and Saetta, 2012).

Definition 14. (**Required information**) Let $TCRM = (Acts, C_T, Res)$ be a TConDec-R process model and $PI^{id} = (TCRM, \sigma^{id})$ be a related process instance. Then:

The **required information** of process instance PI^{id}, $ReqInf^{id} = (EST^{id}, ReqInfActs^{id})$, can be defined as the information that is required related to.

- the expected start time of process instance PI^{id} (i.e., EST^{id}) and
- whether or not the activities of process instance PI^{id} need to be executed, i.e., $ReqInfActs^{id}$, where $ReqInfActs^{id} = \{(a_i^{id}, Label), a \in Acts, i \in [1, MAXNT(ra^{id})]\}$, and $Label \in \{Yes, Not, Unknown\}$.[7] Note that the execution of each activity is either needed (i.e., *Yes*) or not needed (i.e., *Not*) or unknown at the moment (i.e., *Unknown*).

Example 2. (**Required information**) The patient corresponding to process instance PI^{id} is expected to be admitted to the WH on Monday at 10:30am. Whether or not this patient requires a Doopler examination (related to process activity $Ex12_1^{id}$, cf. Table 3) is usually known after meeting the anesthetist, i.e., some time before actually performing the examination. Accordingly, we obtain $ReqInf^{id}$ = (Monday10:30am, $ReqInfActs^{id}$), ($Ex12_1^{id}$, *Unknown*) $\in ReqInfActs^{id}$.

5. Generating optimized enactment plans under rolling planning horizons

This section details the proposed approach. Section 5.1 provides preliminary definitions and setup parameters required to explain how planning under rolling planning horizons is performed (cf. Section 5.2).

5.1. Preliminary definitions

To be able to support scenarios in which the enactment of process instances starts continuously over time, we propose the generation of optimized enactment plans on a rolling horizon basis with a finite planning horizon (Karimi et al., 2003; Tiacci and Saetta, 2012) (cf. Fig. 5). At a specific planning point T, some process instances need to be considered for generating the optimized plan, i.e., the ones that are (or are expected to be) executed at any moment during the planning horizon (cf. process instances PI1 to PI8 in Fig. 5). For these process instances there may be known as well as unknown required information (cf. Def. 14) that needs to be accessed (known information) or forecasted (unknown information). The forecasts are provided based on previously executed process instances in the considered scenario, similar to the mechanisms described in Zhao et al. (1995), Tiacci and Saetta (2012).

Planning at T implies making decisions for all activities related to instances that have been already started as well as instances to be started in close future. Those activities which are expected to start before the end of the *fixed timeframe* (*FT*, cf. Def. 15) are fixed for future plannings, i.e., they will not be replanned (cf. black rectangles in Fig. 5). This way

[7] As mentioned in Section 3.2, $MAXNT(ra^{id})$ refers to the upper bound of the domain of CSP variable $ra^{id}.nt$.

we can avoid that the activities, which are planned to be executed in near future, are replanned at future planning points.

Definition 15. (**Fixed timeframe** & **Fixed timeframe length**) Let T be a planning point. Then: The **fixed timeframe** *FT* defines the temporal window that embraces the activities of an enactment plan which are considered being confirmed activities (i.e., these activities will not be replanned at future planning points). Note that such temporal window starts at T and finishes at T + FTL, with *FTL* being the **fixed timeframe length**.

There are activities that are, although planned at *T*, free to be replanned at future planning points as they are supposed to start after the considered fixed timeframe, i.e., after $T+FTL$ (see the lined rectangles in Fig. 5). In the proposed approach, we consider a sequence of planning points according to a replanning periodicity (RP, cf. Def. 16). That is, each successive planning point differs from the previous one in *RP* time units.

Definition 16. (**Replanning periodicity**) The **replanning periodicity** *RP* states the time spent between two successive planning points when performing planning under rolling planning horizons.

This way, at planning point T + RP, process instances PI1 to PI10 in Fig. 5 are considered as all of them are inside the forward rolled planning horizon. At T + RP, some of the previously planned activities may have to be replanned (see the lined rectangles in Fig. 5), whereas other activities are planned for the first time (see the white rectangles in Fig. 5). In the scenario from Fig. 5, the activities planned for the first time are the ones related to PI9 and PI10. Note that these two process instances are supposed to start after T + PHL, but before T + RP + PHL.

5.2. Planning under rolling planning horizons

The proposed approach for performing planning under rolling planning horizons is detailed in Alg. 1 and illustrated in Fig. 6. For performing such planning, the required input data includes: (a) the TConDec-R process model (cf. Def. 2), (b) the initial planning point (cf. Def. 12), (c) the fixed timeframe length (cf. Def. 15), (d) the replanning periodicity (cf. Def. 16), and (e) the planning horizon length (cf. Def. 13).

First, the initial COP-TConDec-R problem as well as the initial plan are generated (cf. Lines 1–3 of Alg. 1, cf. Alg. 2). For this, the process instances considered when generating the optimized enactment plan (i.e., the ones that are or are expected to be executed during the planning horizon) are retrieved (cf. Line 1 of Alg. 2). Following this, the required information on these instances (cf. Def. 14) is accessed or forecasted (cf. Line 2 of Alg. 2). Then, the COP-TConDec-R problem is derived from the TConDec-R process model, required information, and current event log (that is initially empty), as explained later (cf. Line 3 of Alg. 2). Afterwards, the optimized enactment plan is generated by considering the previously created COP-TConDec-R problem, the current planning point, the defined fixed timeframe length (that takes value 0 when generating the initial plan), and the information related to the considered process instances (cf. Line 4 of Alg. 2).

Algorithm 1: Generating optimized enactment plans under rolling planning horizons

input : TConDec-R process model *tcrm*, Planning point *t*, Fixed timeframe length *ftl*, Replanning periodicity *rp*, Planning horizon length *phl*

```
1  COP-TConDec-R problem p
2  EnactmenPlan ep
3  (p, ep) ← generateInitialPlan(tcrm, t, phl)
4  repeat
5  |  waitUntilReplanningIsRequired(rp, ep, p, tcrm, t)
6  |  t ← getCurrentTime()
7  |  EventLog el ← getCurrentEventLog(t)
8  |  tcrm ← checkForModelUpdates(tcrm, el)
9  |  (p, ep) ← generateReplanning(tcrm, t, ftl, phl, p, el, ep)
10 until;
```

Algorithm 2: generateInitialPlan

input : TConDec-R process model *tcrm*, Planning point *t*, Planning horizon length *phl*

output: COP-TConDec-R problem *newP*, EnactmenPlan *newEp*

```
1  Set<ProcessInstance> currentPIs ← processInstances(tcrm, t, phl)
2  Set<RequiredInformation> reqInf ← Information(currentPIs)
3  newP ← createCSP(tcrm, reqInf, ∅)
4  newEp ← solveCSP(p, t, 0, currentPIs, reqInf)
```

Algorithm 3: generateReplanning

input : TConDec-R process model $tcrm$, Planning point t, Fixed timeframe length ftl, Planning horizon length phl, COP-TConDec-R problem p, EventLog el, EnactmenPlan ep
output: COP-TConDec-R problem $newP$, EnactmenPlan $newEp$

```
1 Set<ProcessInstance> currentPIs ← processInstances(tcrm, t, phl)
2 Set<RequiredInformation> reqInf ← Information(currentPIs)
3 if modelOrPlanDeviates(tcrm, el, ep) then
4 |   newP ← createCSP(tcrm, reqInf, el)
5 else
6 |   newP ← p
7 newEp ← solveCSP(newP, t, ftl, currentPIs, reqInf)
```

When proceeding with the execution of the process, it becomes necessary to check whether replanning is required (cf. Line 5 of Alg. 1). Replanning might be required when either (1) reaching a replanning point or (2) a deviation from the optimized enactment plan occurs in the current execution. To cope with the latter, progress of the process as well as the resource availabilities are monitored during process enactment. If certain events occur (e.g., activities get started/completed or resources become (un) available) the event log is updated accordingly. Whenever updating the event log, the Replanning Module analyzes the previously generated optimized plan as well as the events related to process enactment. In particular, the Replanning Module checks whether the current enactment matches the plan. If this does not apply, replanning will be required. In general, the latter becomes necessary to cope with (1) users deviating from the original plan, (2) incorrect estimates (e.g., activity enactments taking longer or shorter than estimated), and (3) changes regarding resource availability (e.g., a resource might become unavailable during runtime). Note that not every deviation requires replanning. In particular, no replanning becomes necessary if there is enough slack time between activities to withstand the deviation without invalidating the enactment plan, i.e., the amount of time such activities can be delayed without causing another activity to be delayed or impacting the completion time of the enactment plan.

In case replanning is required, the current event log is retrieved (cf. Line 7 of Alg. 1) to check whether the TConDec-R process model needs to be updated (cf. Line 8 of Alg. 1). The latter becomes necessary when the estimates of activity durations or resource availability are changed unexpectedly. Then, the replanning is performed (cf. Line 9 of Alg. 1).

For the replanning step (cf. Alg. 3), the following input data is required: (a) the TConDec-R process model (cf. Def. 2), (b) the planning point (cf. Def. 12), (c) the fixed timeframe length (cf. Def. 15), (d) the planning horizon length (cf. Def. 13), (e) the previously created COP-TConDec-R problem (cf. Def. 10), (f) the current event log (cf. Def. 5), and (g) the current enactment plan (cf. Def. 11). As output of Alg. 3, the updated COP-TConDec-R problem as well as the updated optimized enactment plan are generated. In Algorithms 2 and 3, first of all, the process instances that are or are expected to be executed during the planning horizon are retrieved (cf. Lines 1–2 of Alg. 3). In case replanning became necessary due to a substantial deviation (i.e., not just because reaching a replanning point) (cf. Line 3 of Alg. 3), it must be checked whether the previously generated COP-TConDec-R problem needs to be updated (cf. Line 4 of Alg. 3). Note that the estimates, initially used for generating the COP-TConDec-R problem, might have to be updated as well. As last step, the updated enactment plan is generated (cf. Line 7 of Alg. 3).

The *createCSP* function is based on our previous work for generating optimized enactment plans from TConDec-R process models (Barba et al., 2012; Jiménez-Ramírez et al., 2018). Therefore, the COP-TConDec-R problem is initially created as explained in Def. 10. However, several extensions are required to cope with the challenging rolling planning horizons. To be more precise, the initial COP-TConDec-R problem is extended as follows:

- For each process instance PI^{id},

  - the start time of all corresponding activities must be greater or equal to the expected start time of PI^{id} (i.e., EST^{id}). This requires the addition of constraints to the CSP (i.e., $a_i^{id}.st \geqslant EST^{id} \mid a_i^{id} \in ra^{id}.sacts$, $ra^{id} \in RActs$),
  - the CSP variable *sel* of all corresponding activities must be set according to the required information of the process instance.

  $ReqInfActs^{id} = \{(a_i^{id}, Label)\}$ (i.e., $a_i^{id}.sel = f(Label) \mid \{a_i^{id} \in ra^{id}.sacts,\ ra^{id} \in RActs\}$), with:

$$f(Label) = \begin{cases} 0 & \text{if } Label = \text{“}Not\text{”} \\ 1 & \text{if } Label = \text{“}Yes\text{”} \\ \{0,1\} & \text{otherwise} \end{cases}$$

- At planning point T,

  - all CSP variables related to events are considered as fixed with the value recorded in the event log, and
  - all temporal CSP variables (i.e., *st* and *et*), which are instantiated to values that are placed within the temporal window $[T, T + FTL]$, are considered fixed according to the fixed timeframe length (cf. Def. 15).

In Jiménez-Ramírez and Barba (2018), we formalized TConDec-R constraints based on a catalogue of well-known global constraints. Moreover, the equivalent low-level constraints of this formalization were provided. Based on this information, two equivalent implementations can be obtained, one based on global constraints and another one based on low-level constraints. The implementation of the proposed approach and, therefore, the generation of optimized enactment plans, is based on global constraints. In this respect, given a TConDec-R model, we used the implementation based on low-level constraints to ensure that the generated plans fulfil all low-level constraints related to the TConDec-R constraints of such a model.

6. Improving process support through optimized enactment plans

The proposed approach focuses on flexible scenarios modeled in terms of a declarative process modeling language (i.e., TConDec-R). On one hand, declarative languages offer a high degree of flexibility; on the other, executing a declarative model entails larger efforts for users compared to imperative models (Haisjackl et al., 2016). With the goal of

supporting users in such a flexible context, we propose the generation of optimized enactment plans based on TConDec-R specifications. Such plans enable process support (cf. Fig. 7), e.g., predicting enactment times of future activities (cf. Section 6.1) Jiménez-Ramírez et al., 2018), generating personal schedules (cf. Section 6.2), and recommending appointments to process stakeholders (e.g., customers or patients) (cf. Section 6.3).

6.1. Time prediction

There exist many process scenarios for which time is of utmost importance (Westergaard and Maggi, 2012) and, hence, reliable time predictions are crucial for any PAIS (van der Aalst et al., 2011). For example, in collaborative scenarios, it is often required to negotiate and agree upon future enactment times. In particular, such prediction should not be based on a single process instance in isolation, but consider all process instances and resources being concurrently executed (Schellekens, 2009). The optimized enactment plans generated by the proposed approach enable this use case as time information becomes available based on the estimated duration of the activities (Jiménez-Ramírez et al., 2018). For a given state of a particular process instance, the expected completion time of this instance and its activities can be calculated based on the expected end time of the remaining activities of the respective optimized enactment plan.

Definition 17. (**Time prediction**) Let $TCRM = (Acts, C_T, Res)$ be a TConDec-R process model (cf. Def. 2), $\#inst$ be the number of planned process instances, $RActs$ be the set of repeating activities related to $TCRM$ (cf. Def. 9), $EP = \{(ra^{id}.nt' | ra^{id} \in RActs) \cup (a_i^{id}.st', a_i^{id}.et', a_i^{id}.res', a_i^{id}.sel', \text{with } a_i^{id} \in ra^{id}.sacts\}$ be an enactment plan related to $TCRM$ (cf. Def. 11), activity a_i^{id} be a specific process activity of $TCRM$, and id be a specific process instance of $TCRM$. Then: The **time prediction** related to (EP, a_i^{id}, id) consists of the tuple $(a_i^{id}, a_i^{id}.st', a_i^{id}.et')$ meaning that for process instance id, activity a_i^{id} is expected to start at time $a_i^{id}.st'$ and expected to finish at time $a_i^{id}.et'$.

Example 3. (**Time Prediction**) Regarding the plan depicted in Fig. 7a, activity A_1^1 (i.e., first execution of activity A of process instance 1) is expected to be started at 8am and be completed at 8.50 pm (cf. Fig. 7b1).

With the proposed approach, predictions of resource allocations also become possible as resource allocations can be observed in the plan as well.

Generally, predictions are quite reliable as they consider global process information on multiple instances as well as cross-instance constraints. Finally, predictions may be continuously updated during process enactment through replanning considering the actual enactment status of the process.

The ability to predict the start/completion times of both activities and process instances based on optimized enactment plans offers several advantages. First, constraint violations can be foreseen and avoided by detecting inconsistencies in models prior to as well as during their enactment. Second, coordination in collaborative scenarios, where contracts among multiple parties need to be adhered to, can be improved as realistic deadlines can be established and response times be optimized.

6.2. Personal schedules

Usually, the actors involved in process enactment are unaware of the next activities they shall perform as long as these activities have not become enabled (i.e., all preceding activities have been completed Eder et al., 2003). As users do not have any information on upcoming activities, they cannot plan their work ahead. With the proposed approach this drawback is overcome by generating a set of personal schedules based on the given optimized enactment plan (Eder et al., 2003).

Definition 18. (**Personal schedule**) Let $TCRM = (Acts, C_T, Res)$ be a TConDec-R process model (cf. Def. 2), $\#inst$ be the number of planned process instances, $RActs$ be the set of repeating activities related to $TCRM$ (cf. Def. 9), $EP = \{(ra^{id}.nt' | ra^{id} \in RActs) \cup (a_i^{id}.st', a_i^{id}.et', a_i^{id}.res', a_i^{id}.sel', \text{with } a_i^{id} \in ra^{id}.sacts\}$ be an enactment plan related to $TCRM$ (cf. Def. 11), and w be a specific actor that represents a resource of $TCRM$. Then: A **personal schedule** related to enactment plan EP and actor w consists of a set of tuples $\{(a_i^{id}, a_i^{id}.st', a_i^{id}.et') | a_i^{id}.res' = w\}$ meaning that w shall start the enactment of activity a_i^{id} of process instance id at time $a_i^{id}.st'$, and this enactment is expected to be finished at time $a_i^{id}.et'$.

Example 4. (**Personal schedule**) Consider Fig. 7b2). The personal schedule of actor $w1$ corresponds to $\{(A_1^1, 8.00, 8.50), (A_3^1, 8.50, 9.40), (D_2^1, 9.40, 10.50), (A_7^1, 10.50, 11.40), (B_6^1, 11.40, 13.20)\}$.

Equipping actors with personal schedules based on optimized enactment plans allows:

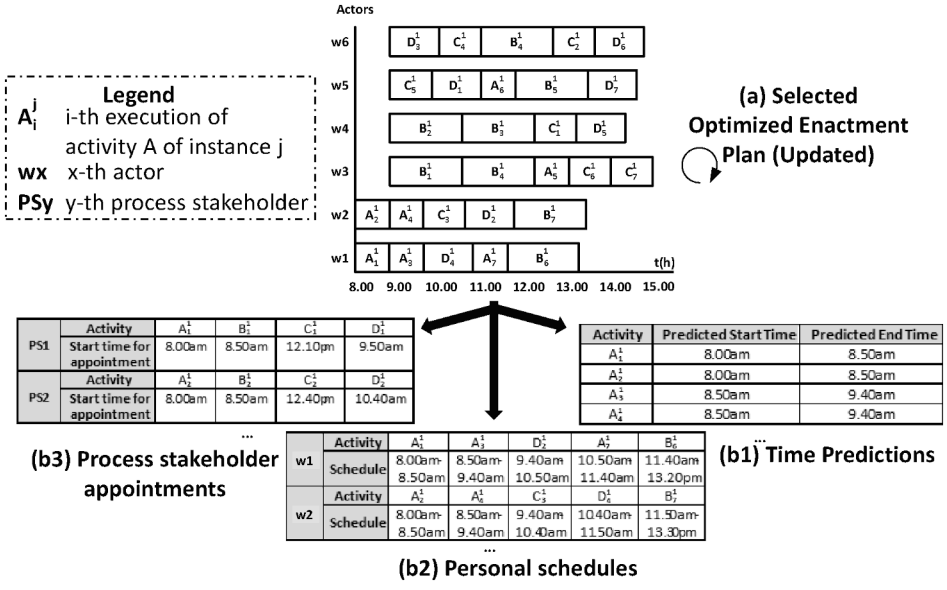

Fig. 7. Optimized enactment plans for improving process support.

Table 4
Evaluation design.

(a) Research question	
Title	*Description*
RQ$_1$: How does the proposed approach behave when considering different settings for the rolling planning horizon parameters in a real-world scenario?	This research question is related to how the rolling planning horizon parameters affect the quality of the generated optimized enactment plans as well as the quality of resource allocations.
(b) Independent variables	
Description	*Values*
FTL: Length of the fixed timeframe (days)	$\{2, 5, 10, 15, 20\}$
RP: Replanning periodicity (days)	$\{1, 2, 3\}$
(c) Response variables	
Description	
StayLength: Average duration (in days) of patient stays in WH before the surgery	
%Used: Percentage of time each resource is used in relation to its total availability	

1. reducing the retention period of activities in worklists before they are started as the actor already knows the work items lying ahead,
2. avoiding latency or deadline violations as all generated plans meet the constraints imposed by the declarative specification,
3. completing the enactment of the process, while not violating any constraint and optimizing given objective functions, and
4. organizing work better within working hours as the optimized plans are generated in a way such that all activities are scheduled within a specific timeframe, i.e., if actors follow their schedule, they will probably finish their working day at the expected time.

For each actor involved in the execution of a given enactment plan, there exists exactly one optimized personal schedule. For example, as the enactment plan of Fig. 7a involves 6 actors (i.e., $w1 - w6$), there are 6 personal schedules.

Each time the optimized plan is updated during process enactment due to a replanning, the personal schedules of all actors need to be updated according to the resulting plan. Note that when generating a new plan through replanning, the partial enactment traces of the respective instances (i.e., the activities started or completed by the respective actors) are considered (i.e., migrated to the new plan). Consequently, the newly generated plans are always compliant with those traces as the current process enactment state is reflected in the new plans. In this way, the adaptation of the personal schedules to the newly selected plan can be addressed correctly.

6.3. Process stakeholder appointments

In many environments, the enactment of processes entails the coordination with process stakeholders (e.g., customers or patients). Consequently, bad planning usually causes several shortcomings: (1) process stakeholder appointments can be only announced very late, (2) waiting times of process stakeholders are increased, and (3) frequent constraint violations might force certain activities to be aborted and rescheduled later. All these shortcomings are annoying for process stakeholders, cause unnecessary costs and efforts, and additionally threaten future appointments.

To overcome these problems, we propose the use of optimized enactment plans for suggesting process stakeholder appointments.

Definition 19. (**Process stakeholder appointment**) Let $TCRM = (Acts, C_T, Res)$ be a TConDec-R process model (cf. Def. 2), $\#inst$ be the number of planned process instances, $RActs$ be the set of repeating activities related to $TCRM$ (cf. Def. 9), $EP = \{(ra^{id}.nt' \mid ra^{id} \in RActs)\} \cup \{(a_i^{id}.st', a_i^{id}.et', a_i^{id}.res', a_i^{id}.sel') \text{, with } a_i^{id} \in ra^{id}.sacts\}$ be an enactment plan related to $TCRM$ (cf. Def. 11), and id be the identification of a specific process stakeholder that represents an instance of $TCRM$. Then:

Table 5
Additional run-time information for the activities.

ID	CritRes	CompWith
Ex0	Staff	–
Ex1	Equipment	Ex12
Ex2	Equipment	–
Ex3	Equipment	–
Ex4	Equipment	–
Ex5	Equipment	–
Ex6	Equipment	–
Ex7	Equipment	–
Ex8	Equipment	Ex9
Ex9	Equipment	Ex8
Ex10	–	–
Ex11	Equipment	–
Ex12	Equipment	Ex1
Ex13	Staff	–
Ex14	Staff	–
AN	–	–
SU	Equipment	–

Table 6
Resource availabilities.

Role	Activities requiring that role	Av. res
StaffEx0	Ex0	2
EquipEx1Ex12	Ex1 and Ex12	1
EquipEx2	Ex2	1
EquipEx3	Ex3	1
EquipEx4	Ex4	2
EquipEx5	Ex5	1
EquipEx6	Ex6	1
EquipEx7	Ex7	1
EquipEx8Ex9	Ex8 and Ex9	2
EquipEx11	Ex11	1
StaffEx13	Ex13	1
StaffEx14	Ex14	1
EquipSU	SU	2

The **process stakeholder appointments** related to (EP, id) consist of a set of tuples $\{(a_i^{id}, a_i^{id}.st')\}$ meaning that activity a_i shall start at time $a_i^{id}.st'$ for process stakeholder id.

According to Def. 19, for each activity of any process instance, an appointment is generated for the corresponding process stakeholder taking the start time of the activity into account.

Example 5. (**Process stakeholder appointments**) Consider the enactment plan depicted in Fig. 7a according to which the appointments $\{(A_1, 8.00), (B_1, 8.50), (C_1, 12.10), (D_1, 9.50)\}$ will be suggested to process stakeholder 1 (cf. Fig. 7b3).

After replanning, the updated plan is compliant with the partial traces of the respective process instances (cf. Section 6.2). Hence, the process stakeholder appointments can be always adapted correctly to the new plan. However, changing appointments that have already been communicated to the process stakeholders implies the re-notification of the latter. Usually, it is desirable to avoid changes in stakeholder appointments if possible. This can be accomplished by considering the minimization of changes regarding previous plans as an additional objective function when generating the enactment plans.

7. Evaluation

As motivated in Section 1, the primary research question (PRQ) addressed in our work is as follows: *Is the proposed approach useful for coping with real process scenarios in which the enactment of process instances starts continuously over time?* To answer *PRQ*, a more specific research question is investigated in the following (i.e., RQ_1 in Table 4). Section

Table 7
Experimental results regarding the quality of the generated plans.

FTL	RP	#Acts	#Constraints	StayLength
2	1	161,23	1343,37	41,3
2	2	173,37	1436,77	41,8
2	3	186,96	1572,16	42,2
5	1	126,81	1051,72	36,6
5	2	135,98	1132,55	37,1
5	3	139,45	1163,25	36,8
10	1	80,56	673,23	31,9
10	2	87,43	741,04	32,3
10	3	110,55	921,25	33,4
15	1	73,35	609,68	31,9
15	2	75,38	631,40	32,3
15	3	85,13	714,02	33,2
20	1	69,69	578,21	31,7
20	2	70,92	583,45	32,1
20	3	75,52	627,52	32,3

7.1 motivates why the scenario introduced in Section 3.3 is selected for demonstrating and evaluating our approach. Section 7.2 details additional run-time information about the scenario, which is required to apply the proposed approach. Section 7.3 explains the study design. Section 7.4 describes the data collection, while Section 7.5 analyzes the obtained results. Finally, Section 7.6 discusses the plan validity and Section 7.7 the lessons learned.

7.1. Case selection

We evaluate our approach by applying it to a particularly challenging healthcare process scenario, i.e., the diagnostic and therapeutic procedures required for treating patients with ovarian carcinoma in a Woman Hospital (Schultheiß et al., 1996; Ovarian cancer (CG122), 2011). This scenario is selected as it is the most challenging one of the healthcare scenarios we analyzed. Moreover, this scenario is well suited as (1) it presents (a) a flexible nature, (b) (cross-instance) constraints related to the time and resource perspectives, and (c) the need of an efficient management of shared resources according to an optimization criterion, (2) it can be modelled with TConDec-R (cf. Fig. 4), and (3) it allows estimating activity durations, resource availability, and the number of process instances being started within a specific timeframe.

Note that we did not consider the presence of repeating activities as a strong requirement for the scenario selection since in our previous work (Jiménez-Ramírez et al., 2018) we already checked the correctness and efficiency of the proposed constraint-based approach when managing processes with activities that are performed multiple times per process instance. In particular, repeating activities are supported by the proposed formalization and have been considered since we have initially proposed the TConDec-R process modeling language.

7.2. Run-time information for the scenario

To be able to apply the proposed approach to the scenario additional run-time information, which has not been considered in Section 3.3, is required (cf. Table 5): column *ID* contains the identifier of the activity, column *CritRes* indicates whether the critical resource for performing the respective activity is staff or medical equipment, and column *CompWith* contains other examinations requiring the same critical resource and, hence, competing with the present activity for this limited resource. According to Table 5, there are 13 types of critical resources (i.e., 13 roles), with restricted availabilities in the considered scenario (cf. Table 6). Regarding resource *EquipSU*, there are 2 operating theatres in the Women's Hospital (with respective teams), which are run 5 days per week and 8 h per day (6,5 h on Friday). These two operating theatres not only deal with the considered surgeries, but with other kinds of surgeries (e.g., minimum invasive surgery, mastectomy, breast ablation) as well. The two operating theatres are usually available to deal with the considered surgeries except in about 15–20% of the working days of a year in which only 1 operating theatre is available to deal with the considered surgeries. Note that the number of available non-critical resources is not relevant in the considered scenario.[8]

7.3. Design

The measures we use for investigating research question RQ_1 (cf. Table 4), i.e., the response variables, are explained in Table 4.

In order to answer RQ_1, we consider two response variables.

1. The *average duration (in days)* of patient stays in the Women's Hospital before the surgery takes place – this information is directly related to the quality of the generated optimized enactment plans. The shorter the patient stays in WH are, the higher the quality of the plans is.
2. The *percentage of time each resource is used in relation to its total availability* is related to the quality of resource allocation in the generated plans. The greater the percentage of time each resource is used become, the higher is the quality of resource allocation.

Initially, over a period of 15 days, patient treatment performance in the WH is simulated, considering real data with the goal of mimicking the ordinary state of the WH. Then, the approach is applied over a period of 2 months. In the given scenario, the planning horizon length (PHL) is fixed to one week as the information about patients arriving at the WH is known one week in advance. To check the behavior of the proposed approach when facing problems of different complexity, results are generated considering different values for the length of the fixed timeframe *FTL* (cf. Def. 15) and for the replanning periodicity *RP* (cf. Def. 16) as detailed in Table 4. This way, the proposed approach is tested over 15 different configurations. Moreover, to average results over a collection of diversified datasets, 30 case datasets are randomly generated. The response variables are then calculated by considering the average values, which are obtained for the tests related to the 45 intermediate days. Note that the results obtained for the first and the last fifteen days are discarded as the goal consists of analyzing the behavior of the approach in the ordinary WH state (i.e., when there is already an ordinary number of patients staying in the WH, while other patients are arriving).

As machine, we use an Intel Core i7, 3 GHz, 6 GB memory running on Windows 7. For the experiments, the search algorithm is run until reaching a 300-s CPU time limit. To implement the constraint-based problems and algorithms, we use CPLEX, which is able to generate high-quality solutions for highly constrained problems in an efficient way. CPLEX provides a scheduling module offering high-level constraint modeling as well as search abstraction, of which both are specific to scheduling.

7.4. Data collection

Table 7 shows the average number of activities to be planned (i.e., *#Acts*) and the number of considered constraints (i.e., *#Constraints*) to solve the problem as well as the average duration (in days) of patient hospital stays (i.e, *StayLength*) for each specific problem (i.e., stated by pair $\{FTL, RP\}$). Furthermore, Fig. 8 shows the average percentage of time each resource is used (compared to its overall availability) against *FTL* and *RP*, whereas Fig. 9 depicts the minimum, average, and maximum percentage of resource utilization to show which resources are most critical. As detailed in Section 7.2, the critical resource for performing each activity in the considered scenario can be staff or medical equipment.

[8] An example of the different processes involving 14 patients for the considered scenario is depicted in https://doi.org/10.5281/zenodo.4387200.

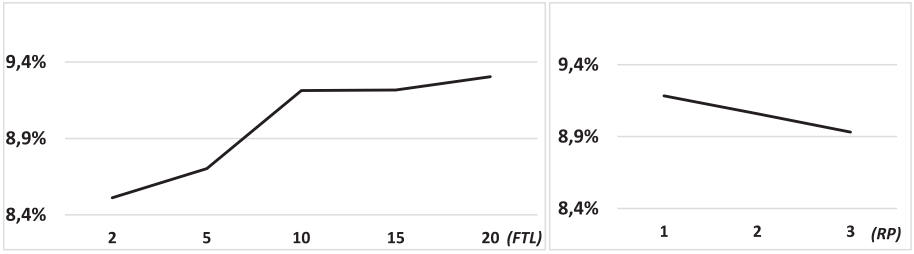

Fig. 8. Average percentage of use of resources (%*Used*) against *FTL* (left) and *RP* (right).

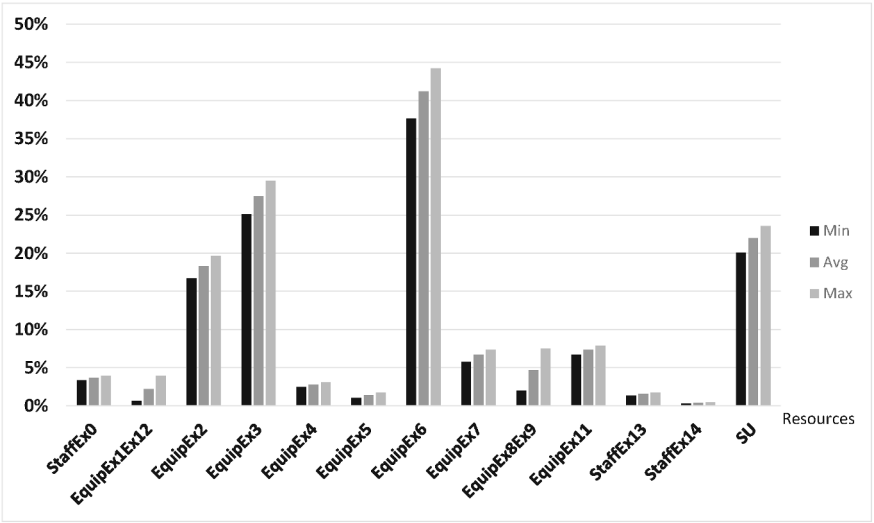

Fig. 9. Minimum, average, and maximum percentage of time each resource is used when compared to its overall availability (%*Used*).

7.5. Analysis

In the following, the case study findings are interpreted with the goal of answering research question RQ_1.

As can be observed in Table 7, the greater the length of the fixed timeframe (*FTL*) becomes the more activities are fixed and, therefore, the fewer decision variables of the related COP-TConDec-R problem need to be instantiated (cf. column *#Acts* in Table 7). On one hand, this reduces the complexity of the problem and, hence, allows for a better optimization when setting a time limit. On the other, the fewer decision variables are free to be instantiated, the fewer improvements can be performed through replanning and, hence, the lower optimizations become. To obtain good results for a given time limit, *FTL* must be high enough to reduce the complexity of the problems to be solved; at the same time, *FTL* must be small enough to allow for improvements over the plans. As shown in Table 7, for the considered scenario, the average duration of patient hospital stays decreases noticeable as *FTL* increases until reaching value 10. An additional increase of *FTL* seems to have no influence on the quality of the plans obtained. This can be explained with the fact that the plans obtained for $FTL = 20$ are rather good and similar to the ones obtained after freeing more variables (i.e., for $FTL = 10$). Hence, 10 seems to be the best *FTL* value as this is the one with the greatest number of free decision variables.

As another aspect, the schedules of patients and staff become more stable as *FTL* increases. Therefore, *FTL* should be established in such a way that high satisfaction of the involved process stakeholders is obtained. Additionally, for the same *FTL*, the quality of the found plans slightly increases since *#Acts* and *#Constraints* decrease due to the decrease of *RP*. Note that this makes sense –the more often replannings are performed, the fewer activities should be replanned and the greater the probability of improving plans becomes. Moreover, the considered problems are highly constrained entailing a great complexity. When applying the approach to a real-world scenario, only one case needs to be managed at each planning point and, therefore, the time limit could be much greater than the one established for this evaluation.

As shown in Fig. 8, *FTL* seems to be much more influential than *RP*. This makes sense as the percentage of time a resource is used in relation to its total availability increases with growing plan quality. In Fig. 9 the depicted values are related to the percentage of times the corresponding examinations are requested and the duration of these examinations. As a result, we can conclude that the most critical resource is EquipEx6 (note that EquipEx6 is required for performing Ex6, and the latter examination is requested for all patients and lasts 2h15min, cf. Table 3). In turn, StaffEx14 is used least often as it is required for performing Ex14, which is only requested for 10% of the patients and lasts only 30 min (cf. Table 3).

Note that the scenario is highly constrained and the constraints do not allow for a high usage of certain resources. To be more precise, cross-instance constraints greatly influence the use of certain resources, whereas time-based exclusive constraints forbid the execution of certain activities for a certain period of time regardless of resource availabilities. Furthermore, there are resources only required for a low percentage of cases, e.g., StaffEx13 is required for performing Ex13, which lasts 60 min, but is requested only for 20% of the patients, while its availability is 10 h a day.

Considering all this information, RQ_1 can be answered as follows (1) *FTL* seems to be much more influential than *RP* regarding the quality of both the generated optimized enactment plans and the allocated resources; (2) For the same *FTL*, the quality of the found plans slightly increases when *RP* decreases; (3) To obtain good results for a given time limit, *FTL* must be high enough to reduce complexity of the problems to be solved and small enough to allow for improvements over the plans;

(4) For the considered scenario, 10 seems to be the best *FTL* value; and (5) The schedules of patients and staff become more stable with increasing *FTL*.

7.6. Threats to validity

Regarding the construct validity of the presented evaluation, we have learned that the proposed plan is suitable for reaching the intended goal (i.e., analyzing the behavior of the approach in the context of a sophisticated process scenario from the real world). To be more precise, the proposed response variables and the generated cases are considered as appropriate for answering research question RQ_1. This way, the values obtained for the response variables are analyzed with the goal of checking the behavior of the proposed approach in a real-world scenario (cf. Section 7.5). However, additional response variables and experiments could be defined to broaden the analysis as well as the related findings.

Regarding external validity (i.e., the domains to which study findings may be generalized), the study findings can be generalized to scenarios that (1) can be modelled with TConDec-R, (2) allow estimating activity durations, resource availability, and the number of process instances being started within a specific timeframe, and (3) present (a) a flexible nature, (b) (cross-instance) constraints related to the time and resource perspectives, and (c) the need for an efficient management of shared resources according to an optimization criterion.

We are aware that our empirical evaluation is based on the simulation of a real-world process scenario, but was not performed in a real environment. The latter could reveal practical issues, like missing constraints, and even constraints not covered by TConDec-R yet. We admit that this constitutes a limitation, and we are aware that the evaluation of the proposed approach in a practical setting would be desirable to make results more realistic. However, the study findings may be generalized to other scenarios presenting characteristics similar to the ones of the Women Hospital scenario. Evaluating the proposed approach with another real process scenario (that shows different features, e.g., activities performed multiple times per process instance) would further generalize our study findings.

7.7. Lessons learned

As empirically demonstrated, the quality of both the generated optimized enactment plans and the resource allocation largely depends on the selection of the rolling planning horizon parameters. To be more precise, *FTL* seems to be much more influential than *RP*. As an additional finding, it could be observed that the stability of the schedules of both patients and staff, which is considered as crucial in the scenario, mostly depends on the *FTL* parameter.

We can conclude that the evaluation goals are achieved, i.e., the effectiveness of applying the proposed approach in a real and complex scenario as well as the behavior of the proposed approach under rolling planning horizons have been successfully tested and analyzed.

8. Discussion

The current work has been motivated by the needs we discovered when analyzing sophisticated process scenarios from the healthcare domain. When modeling the particularly challenging scenario considered in Section 3.3, we were aware that no process modeling proposal was able to address its requirements, as detailed in our previous work (Jiménez-Ramírez et al., 2018). This led us to propose the TConDec-R language for modeling scenarios that present similar characteristics to the one considered. Moreover, considering the complexity of the scenario constraints as well as the high number of activities, resources and patients to be managed, a more advanced support of ward staff in planning the patients' examinations and surgeries is urgently needed, as detailed in Section 3.3.1. For this purpose, we have proposed a constraint-based approach that focuses on reducing the length of patient stays in the hospital (Jiménez-Ramírez et al., 2018).

To make the proposed approach applicable to the considered scenario, replanning and rolling planning horizons needed to be considered and, hence, our approach needed to be extended in this direction.

Altogether, we carried out an empirical evaluation by performing different simulations over the considered healthcare scenario. We believe that the application of the proposed approach can substantially improve its management by (1) avoiding constraint violations, (2) managing shared resources in an effective way, (3) improving and automating (cross-process) coordination between different departments and even hospitals, (4) reducing waiting times of patients as the schedules become known beforehand, and (5) minimizing the length of patient stays in the hospital (i.e., speeding up the many diagnostic procedures required before surgery). Hence, we believe that the proposed approach can be successfully applied to many other knowledge-intensive scenarios presenting similar characteristics. However, we did not run the approach at the operational level of the Women Hospital, which constitutes a limitation of our evaluation.

The proposed approach has revealed certain other limitations. First, the TConDec-R language was proposed after considering a large number of scenarios, particularly from the healthcare domain, but additional extensions could be required when analyzing further scenarios from other domains. In a similar way, such extensions would be required in the constraint-based approach when generating the optimized enactment plans. Considering the high expressive power of the constraint programming paradigm, however, we are confident that the new constraints could be easily integrated in our approach, i.e., we consider the proposed approach as being extensible.

The generated plans are denoted as *optimized* instead of *optimal* as the generation of optimal plans presents NP-complexity (Garey and Johnson, 1979) and, hence, it is not possible to ensure optimality for all cases. However, our results indicate that the approach allows solving the considered problems in an efficient way, leading to substantial duration reductions of patient stays.

The constraint-based approach has been developed in such a way that its efficiency, which we consider as a key success factor, can be ensured. For example, global constraints are implemented through efficient filtering rules (Jiménez-Ramírez and Barba, 2018). Note that the efficiency of the proposed approach is crucial as we need to enforce a time limit for executing the algorithm that generates the optimized enactment plan. Given a specific time limit, the greater the efficiency of the proposed constraint-based approach is, the higher the probability of finding good solutions (according the objective function defined) becomes. Moreover, no deadlock scenarios will occur later during process execution as the generated plans consider all scenario constraints and, therefore, feasible solutions are always obtained.

As explained, the optimized enactment plans are generated with the goal of supporting users during enactment. We are aware that declarative approaches are usually applied to allow for a high degree of freedom during process execution. Therefore, the generated plans are recommendations on how to proceed, but the user is completely free to decide how to actually execute the process. To be more precise, at any point during the execution of a process instance, users may decide what to do next; i.e., they may select the next activity from the set of enabled activities (i.e., the activities that may be currently started not violating the instance constraints afterwards Reichert and Weber, 2012; Barba et al., 2013). However, to guide the user in optimizing the overall process goals, recommendations (i.e., suggesting to start a specific activity using a specific resource) can be provided by a recommendation system that is based on the optimized enactment plans, as detailed in Barba et al. (2013). Note that the user is not obliged to follow the recommendations, but may select any of the enabled activities, i.e., the flexibility of the constraint-based specification is kept. The deviations between the actual enactment of the process instances on the one hand and the optimized enactment plans on the other are addressed through replanning.

Moreover, there may be errors in the estimates that might invalidate the optimized enactment plans. Again, the deviations between actual enactment of the process instances and optimized enactment plans are addressed through replanning. Not every deviation requires replanning. In particular, no replanning becomes necessary if there is enough slack time between activities to withstand the deviation without invalidating the enactment plan.

It is noteworthy that only few decisions are taken at each planning point, while the other decisions are reconsidered at future planning points. To be more precise, those activities which are planned to start before the end of the fixed timeframe (*FT*) remain fixed for future plannings, i.e., they will not be replanned anymore. The remaining activities, however, may be replanned when reaching at future planning points.

We are aware that factoring out the data perspective of the processes during decision support constitutes a limitation of the proposed approach to specific classes of processes. However, to set a focus, this paper mainly deals with the time perspective and thus complements existing works that have utilized the data perspective for improving decision support (Montali, 2010; Montali et al., 2013; Maggi et al., 2013; Borrego and Barba, 2014; Mertens et al., 2017; Slaats et al., 2013). We plan to co-consider both the time and the data perspective in future work.

Altogether, the proposed approach has implications for practice. We could show that we can model complex real-world scenarios in a declarative manner. Moreover, we could demonstrate that these models, when being used with a constraint-based approach for planning and scheduling process activities, can help organizations to improve their business processes (e.g., substantial reductions of the duration of patient stays). Furthermore, although there are some real-world process scenario modelled in a declarative way (e.g., Debois and Slaats, 2015; Strømsted et al., 2018), to the best of our knowledge, this is the first sophisticated real-world process scenario with complex temporal constraints as well as cross-instance constraints modelled in a declarative way.

9. Summary and outlook

This paper built upon previous work (i.e., generating optimized enactment plans from temporal declarative process models) with the goal of making our approach applicable to a large class of real-world scenarios. To be more precise, we consider scenarios where the enactment of process instances starts continuously over time and propose an approach for planning under rolling planning horizons. To validate the proposed approach we perform an evaluation over a sophisticated process scenario from the healthcare domain. Results indicate that the approach allows solving the considered problems in an efficient way leading to substantial reductions of the duration of patient stays.

We are aware that we need to consider certain aspects to broaden the applicability of the proposed approach. First, we intend to analyze further scenarios from other domains presenting different characteristics to check which extensions would be required to be able to apply the proposed approach to these scenarios as well. Second, we will consider other real-world scenarios for which it is possible to conduct an empirical evaluation in a real environment. Third, we intend to develop a tool based on the proposed approach. Fourth, we intend to consider the data perspective of processes when providing decision support as well as generating different optimized enactment plans at each decision point. We will consider these aspects in future work.

CRediT authorship contribution statement

Irene Barba: Methodology, Investigation, Writing - original draft, Supervision. **Andres Jiménez-Ramírez:** Software, Validation, Formal analysis, Visualization. **Manfred Reichert:** Writing - review & editing, Conceptualization. **Carmelo Del Valle:** Writing - review & editing, Conceptualization. **Barbara Weber:** Writing - review & editing, Conceptualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research has been supported by the Pololas project (TIN2016-76956-C3-2-R) of the Spanish Ministry of the Economy and Competitiveness and NICO project (PID2019-105455 GB-C32) of the Spanish Ministry of the Science and Innovation.

References

Ackermann, L., Schönig, S., Petter, S., Schützenmeier, N. & Jablonski, S. (2018). Execution of multi-perspective declarative process models. In Proceedings of OTM (pp. 154–172). Springer.

Barba, I., Lanz, A., Jiménez-Ramírez, A., Weber, B., Reichert, M. & Del Valle, C. (2016). Providing support for the optimized management of declarative processes. In Proceedings of JisBD. Universidad de Salamanca.

Barba, I., Lanz, A., Weber, B., Reichert, M., & Del Valle, C. (2012). Optimized time management for declarative workflows. *Proceedings of BPMDS*, 195–210.

Barba, I., Weber, B., Del Valle, C., & Jiménez-Ramírez, A. (2013). User recommendations for the optimized execution of business processes. *Data & Knowledge Engineering, 86*, 61–84.

Berbner, R., Spahn, M., Repp, N., Heckmann, O. & Steinmetz, R. (2007). Dynamic replanning of web service workflows. In Proceedings of Inaugural IEEE-IES DEST (pp. 211–216). IEEE.

Borrego, D., & Barba, I. (2014). Conformance checking and diagnosis for declarative business process models in data-aware scenarios. *Expert Systems with Applications, 41* (11), 5340–5352.

Borrego, D., Gómez-López, M., & Gasca, R. (2020). Prognosis of multiple instances in time-aware declarative business process models. *Computers in Industry, 120*, Article 103243.

Burattin, A., Maggi, F., & Sperduti, A. (2016). Conformance checking based on multi-perspective declarative process models. *Expert Systems with Applications, 65*, 194–211.

Carvalho, R., Silva, N. C., Lima, R., & Cornlio, M. (2013). Reflex: An efficient graph-based rule engine to execute declarative processes. *Proceedings of IEEE SMC*, 1379–1384.

De Araujo, S., Arenales, M., & Clark, A. (2007). Joint rolling-horizon scheduling of materials processing and lot-sizing with sequence-dependent setups. *Journal of Heuristics, 13*(4), 337–358.

Debois, S. & Hildebrandt, T. (2017). The DCR workbench: Declarative choreographies for collaborative processes. Behavioural types: From theory to tools, river publishers series in automation, control and robotics (pp. 99–124).

Debois, S., & Slaats, T. (2015). The analysis of a real life declarative process. *Proceedings of IEEE SSCI*, 1374–1382.

Eder, J., & Pichler, H. (2005). Probabilistic calculation of execution intervals for workflows. *Proceedings of TIME*, 183–185.

Eder, J., Pichler, H., Gruber, W., & Ninaus, M. (2003). Personal schedules for workflow systems. *Proceedings of BPM*, 216–231.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. New York, NY, USA: W.H. Freeman & Co.

Goedertier, S., Haesen, R., & Vanthienen, J. (2008). Rule-based business process modelling and enactment. *International Journal of Business Process Integration and Management, 3*(3), 194–207.

Haisjackl, C., Barba, I., Zugal, S., Soffer, P., Hadar, I., Reichert, M., Pinggera, J., & Weber, B. (2016). Understanding Declare models: strategies, pitfalls, empirical results. *Software & Systems Modeling, 15*(2), 325–352.

Hildebrandt, T., Mukkamala, R., Slaats, T., & Zanitti, F. (2013). Contracts for cross-organizational workflows as timed dynamic condition response graphs. *The Journal of Logic and Algebraic Programming, 82*(5), 164–185.

Hoenisch, P., Schulte, S., and Dustdar, S. (2013). Workflow scheduling and resource allocation for cloud-based execution of elastic processes. In Proceedings of SOCA (pp. 1–8). IEEE.

Jiang, Y., Xiao, N., Zhang, Y., & Zhang, L. (2016). A novel flexible activity refinement approach for improving workflow process flexibility. *Computers in Industry, 80*, 1–15.

Jiménez-Ramírez, A. & Barba, I. (2018). A constraint-based approach for a declarative temporal business process modeling language. URL:https://doi.org/10.5281/zenodo.4387184. [Online; accessed 23-December-2020].

Jiménez-Ramírez, A., Barba, I. & Del Valle, C. (2018a). A constraint-based approach for managing declarative temporal business process models. In Proceedings of ISD.

Jiménez-Ramírez, A., Barba, I., Del Valle, C., & Weber, B. (2013). Generating multi-objective optimized business process enactment plans. *Proceedings of CAiSE*, 99–115.

Jiménez-Ramírez, A., Barba, I., Fernández-Olivares, J., Valle, C. D., & Weber, B. (2018). Time prediction on multi-perspective declarative business processes. *Knowledge and Information Systems, 57*(3), 655–684.

Jiménez-Ramírez, A., Barba, I., Reichert, M., Weber, B., & Del Valle, C. (2018). Clinical processes – the killer application for constraint-based process interactions. *Proceedings of CAiSE*, 374–390.

Käppel, M., Schützenmeier, N., Schönig, S., Ackermann, L. & Jablonski, S. (2019). Logic based look-ahead for the execution of multi-perspective declarative processes. In Proceedings of EMMSAD (pp. 53–68). Springer.

Karimi, B., Fatemi Ghomi, S., & Wilson, J. (2003). The capacitated lot sizing problem: A review of models and algorithms. *Omega, 31*(5), 365–378.

Lanz, A., Kolb, J., & Reichert, M. (2013). Enabling personalized process schedules with time-aware process views. *Proceedings of CAiSE Workshops*, 205–216.

Lanz, A., Reichert, M., & Weber, B. (2016). Process time patterns: A formal foundation. *Information Systems, 57*, 38–68.

Lanz, A., Weber, B., & Reichert, M. (2014). Time patterns for process-aware information systems. *Requirements Engineering, 19*(2), 113–141.

Maggi, F. (2014). Discovering metric temporal business constraints from event logs. In *Perspectives in business informatics research* (pp. 261–275). Springer.

Maggi, F., Dumas, M., García-Bañuelos, L. & Montali, M. (2013). Discovering data-aware declarative process models from event logs. In Proceedings of BPM (pp. 81–96). Springer.

Maggi, F., & Westergaard, M. (2014). Using timed automata for a Priori warnings and planning for timed declarative process models. *International Journal of Cooperative Information Systems, 23*(1), 1440003.

Mans, R. S., Russell, N. C., van der Aalst, W. M. P., Bakker, P. J. M., & Moleman, A. J. (2010). Simulation to analyze the impact of a schedule-aware workflow management system. *Simulation, 86*(8–9), 519–541.

Marquard, M., Shahzad, M. & Slaats, T. (2015). Web-based modelling and collaborative simulation of declarative processes. In Proceedings of BPM (pp. 209–225). Springer.

Masdari, M., ValiKardan, S., Shahi, Z., & Azar, S. (2016). Towards workflow scheduling in cloud computing: A comprehensive analysis. *Journal of Network and Computer Applications, 66*, 64–82.

Mertens, S., Gailly, F., & Poels, G. (2017). Towards a decision-aware declarative process modeling language for knowledge-intensive processes. *Expert Systems with Applications, 87*, 316–334.

Mertens, S., Gailly, F., & Poels, G. (2019). A generic framework for flexible and data-aware business process engines. In *Advanced information systems engineering workshops* (pp. 201–213). Springer.

Montali, M. (2010). *Specification and verification of declarative open interaction models: A logic-based approach* (Vol. 56). Springer Science & Business Media.

Montali, M., Chesani, F., Mello, P., & Maggi, F. (2013). Towards data-aware constraints in declare. *Proceedings of SAC*, 1391–1396.

Montali, M., Maggi, F., Chesani, F., Mello, P., & van der Aalst, W. M. P. (2013). Monitoring business constraints with the event calculus. *ACM Transactions on Intelligent Systems and Technology, 5*(1), 17.

Oh, J., Cho, N., Kim, H., Min, Y., & Kang, S. (2011). Dynamic execution planning for reliable collaborative business processes. *Information Sciences, 181*(2), 351–361.

Ovarian cancer (CG122) (2011). Ovarian cancer(CG122). URL:http://www.nice.org.uk/CG122. [Online; accessed 13-July-2020].

Pesic, M. (2008). Constraint-Based Workflow Management Systems: Shifting Control to Users. PhD thesis, Technische Universiteit Eindhoven, The Netherlands.

Pesic, M., Schonenberg, M., Sidorova, N., & van der Aalst, W. M. P. (2007). Constraint-based workflow models: Change made easy. *Proceedings of CoopIS*, 77–94.

Reichert, M. & Weber, B. (2012). Enabling flexibility in process-aware information systems. Springer.

Rossi, F., van Beek, P. & Walsh, T. (Eds.), (2006). Handbook of Constraint Programming. Elsevier.

Schellekens, B. (2009). *Cycle Time Prediction in Staffware*. In Master's thesis. Eindhoven: University of Technology.

Schonenberg, M., Weber, B., van Dongen, B., & van der Aalst, W. M. P. (2008). Supporting flexible processes through recommendations based on history. *Proceedings of BPM*, 51–66.

Schönig, S., Ackermann, L., & Jablonski, S. (2018). Towards an implementation of data and resource patterns in constraint-based process models. In *Model-driven engineering and software development* (pp. 271–278). INSTICC, SciTePress.

Schönig, S., Di Ciccio, C., Maggi, F. M. & Mendling, J. (2016). Discovery of multi-perspective declarative process models. In Service-oriented computing (pp. 87–103). Springer.

Schultheiß, B., Meyer, J., Mangold, R., Zemmler, T. & Reichert, M. (1996). Designing the processes for ovarian cancer surgery (in German). Technical Report DBIS-6, University of Ulm.

Slaats, T., Mukkamala, R., Hildebrandt, T., & Marquard, M. (2013). In B. P. M. In Proc (Ed.), *Exformatics declarative case management workflows as DCR graphs* (pp. 339–354). Springer.

Strømsted, R., Marquard, M., & Heuck, E. (2018). Towards low-code adaptive case management solutions with dynamic condition response graphs, subprocesses and data. *Proceedings of EDOCW*, 12–16.

Tiacci, L., & Saetta, S. (2012). Demand forecasting, lot sizing and scheduling on a rolling horizon basis. *International Journal of Production Economics, 140*(2), 803–814.

van der Aalst, W. M. P. (2009). Process-aware information systems: Lessons to be learned from process mining. In Transactions on petri nets and other models of concurrency II (pp. 1–26). Springer.

van der Aalst, W. M. P., Pesic, M., & Schonenberg, M. H. (2009). Declarative workflows: Balancing between flexibility and support. *Computer Science – Research and Development, 23*(2), 99–113.

van der Aalst, W. M. P., Schonenberg, M. H., & Song, M. (2011). Time prediction based on process mining. *Information Systems, 36*(2), 450–475.

van Dongen, B. F., Crooy, R. A., & van der Aalst, W. M. P. (2008). Cycle time prediction: When will this case finally be finished? *Proceedings of CoopIS*, 319–336.

Weske, M. (2019). *Business process management – concepts, languages, Architectures*. Springer.

Westergaard, M., & Maggi, F. (2012). Looking into the future: Using timed automata to provide a priori advice about timed declarative process models. *Proceedings of CoopIS*, 250–267.

Xie, J., Zhao, X., & Lee, T. (2003). Freezing the master production schedule under single resource constraint and demand uncertainty. *International Journal of Production Economics, 83*(1), 65–84.

Xu, H., Pang, J., Yang, X., Yu, J., Li, X., & Zhao, D. (2020). Modeling clinical activities based on multi-perspective declarative process mining with openEHR's characteristic. *BMC Medical Informatics and Decision Making, 20*(303), 1–11.

Zeising, M., Schönig, S., & Jablonski, S. (2014). Towards a common platform for the support of routine and agile business processes. *Proceedings of CollaborateCom*, 94–103.

Zhao, X., et al. (1995). Lot-sizing rules and freezing the master production schedule in MRP systems under demand uncertainty. *International Journal of Production Research, 33*, 2241–2276.

Zugal, S., Soffer, P., Haisjackl, C., Pinggera, J., Reichert, M., & Weber, B. (2015). Investigating expressiveness and understandability of hierarchy in declarative business process models. *Software & Systems Modeling, 14*(3), 1081–1103.