Utilising Bitemporal Information for Business Process Contingency Management

John Wondoh, Georg Grossmann, and Markus Stumptner
School of Information Technology and Mathematical Science
University of South Australia
john.wondoh@mymail.unisa.edu.au
{georg.grossmann, mst}@cs.unisa.edu.au

ABSTRACT

In today's enterprise environment, business processes no longer operate in an isolated fashion, driven purely by human input. Instead, they exchange information across organisations as well as interacting directly with sensors and actuators in the Internet of Things. This means that traditional assumptions about processes having a "perfect" view of the world no longer hold as real world events affect prior plans. Critical decisions must be made based on temporal information of events which potentially lead to reconfiguration of business processes to provide the desired service within specified time limits. Currently, temporal aspects of business processes only consider events in a single time dimension. However, recent architectures such as modern database systems are starting to provide the capabilities for handling two time dimensions. In this paper, we investigate how the impact of a change in an event can be identified using bitemporal information from business events. We take into consideration the underlying data model as well as behaviour specifications in the form of artifact lifecycles. Identifying the impact will enable us to tailor appropriate process adaptions as a result of event time changes.

Keywords

Bitemporal Events; Business Artifacts Lifecycle, Data Model, Impact Analysis, Contigency Management

1. INTRODUCTION

In a competitive market where business organisations must provide quality services to their partners and customers in a timely manner, compliance to temporal requirements is an issue that must be taken seriously. Most business processes in the real world are time constrained such that activities within a process are expected to be completed within some time limit. One of the main problems in business process management (BPM) is the dynamic management of processes to meet these deadlines [17, 11]. Business process models are designed to comply with process specifications as well as temporal requirements of the process. However, during runtime, artifacts in the real-world are prone to changes. These changes may not be reflected in the artifact instances during process execution. In such circumstances, reconfiguring the business process may be required for adapting artifact instances to changes occurring to real-world artifacts.

Considering bitemporal information in business process management can help to reconfigure business processes in a better way. Currently time is viewed in a single dimension by the business process literature, under the assumption that events are captured immediately as they occur [20]. However, in the real world, imperfections result in delays and possibly repeated updates to events that have already been captured [7]. To properly capture time in business process management systems (BPMS), we must view time in two dimensions [19, 7, 10]: the time an event occurs in the real world, which we refer to as *business time*, and the time at which it is captured and responded to by an information system such as a BPMS, which we refer to as *decision time*.[1]

If the relationship of business time and decision time of an event is not properly considered, this may lead to problematic business decisions leading to incorrect business outcomes, poor customer service, or even violation of legal contracts in an inter-organisational setting. For example, a variation in the time of discharge of a hospital patient entitled to home care may result in a period without arranged care if proper precautions are not taken. To resolve such issues requires going through all the temporal information associated with the business process. Therefore, it is important to consider both dimensions of time during decision making to avoid any conflicts later. This may be tedious and error prone. Traditional event processing systems are incapable of handling such events via internal logic and require dedicated implementation of process exceptions, leading to considerable extra effort. Notably, not only must two time dimensions be considered, but also their histories and updates pertaining to a particular event. Novel reasoning capabilities must be introduced to equip BPMSs for managing two time dimensions. In particular, BPMSs must take into account the time an event occurs in the real world, as well as the time it is considered during decision making.

Essentially, accommodating the reality of events occurring differently from when they were predicted to happen

ACE '16 Canberra, ACT Australia

http://dx.doi.org/10.1145/2843043.2843045

[1]This nomenclature is chosen to reflect the business-oriented terminology used by some temporal database system implementations [12]. The temporal database literature [19] often uses the terms "valid time" and "transaction time".

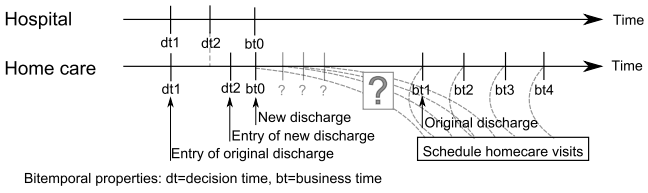

Figure 1: Bitemporal Scenario in Hospital-Home Care Process

requires dynamic adaptation of processes, which means decision making during runtime. In this paper, we present an approach to analyse the impact of changes to bitemporal properties of an event, leading to improved contingency management of business processes. Section 2 provides an example of temporal event handling, and basic definitions. Contributions in the main sections of the paper include:

1. In Section 3, we present an architecture that enables adaptive process execution subject to bitemporal event processing.

2. In Section 4.1, we study the impact of updates of bitemporal events on the data model of a business process.

3. The impact of updates on bitemporal properties of an event on the lifecycle model is studied in Section 4.2.

4. The implications of the impact and a possible reconfiguration approach to resolve resulting issues are dealt with in Section 4.3.

Section 5 discusses related work and compares existing approaches to our approach. We conclude the paper and discuss future work in Section 6.

2. BACKGROUND

In this section, we briefly introduce bitemporal events and decision time as main concepts used in this work.

Consider a scenario where home care is provided by an organisation to patients who are discharged from a hospital. Work in the homecare organisation is planned well in advance. Staff are usually scheduled a week or more in advance. However, once in place, the actual schedules are subject to frequent disruption due to variation in the timing of real world events.

For example, in Figure 1, at time dt_1, a patient is scheduled to be discharged from the hospital at business time bt_1. The homecare organisation schedules home visits to patient from business times bt_1 to bt_4. Assuming the patient is discharged from the hospital at an earlier time bt_0 compared to the initially scheduled time bt_1. In addition, this fact is updated in the hospital system at dt_2, and this may not be immediately communicated to the homecare organisation. Once this becomes known, the homecare organisation is required to plan care visits for the patient for the period between bt_0 and bt_1. A further challenge is that the planning needs to be done within a shorter time period than initially expected.

Both time dimensions of the home care request event, i.e., the planned time (business time) and the time this was registered by the homecare system (system time) must be taken into consideration during decision making. The changes to these time dimensions during execution must be considered in altering the initial decision and adapting the process. Therefore, situations like this require an automatic decision making approach that takes into consideration all temporal dimensions of an event.

Note that with regards to bitemporality, we consider time dimensions to be associated with facts. The latter may be associated with either an event or an object (artifact) in a business process. We focus on events as they can be used to capture the initiation and termination of an activity or process as well as monitor the progress of business processes.

An event is as an occurrence within a particular system or domain, i.e., something that has happened, expected to happen, or is contemplated as having happened in that domain [6]. The properties of an event according to [15] include an *event name* (a name describing the type of event), and a *timestamp* (indicating when the event occurred). In conventional event processing systems, only a single timestamp of an event is captured to determine when the event has occurred. Other temporal dimensions as described in the temporal database literature, e.g., [10, 12, 19] are not exploited. This work focuses on bitemporal events [7].

Definition 1 (Bitemporal Event). *We define as a* bitemporal event *an event e that has associated temporal properties business time bt_e (i.e., the time the event is valid in the real world) and decision time dt_e (i.e., the time when the event is processed in the BPMS). We omit the event e in dt_e and bt_e if e is understood.*

System time as recorded in bitemporal databases must be either past or present time, i.e., it does not extend past the current time, and the same is true in our model for the decision time (the time when an event is processed). Business time on the other hand can be past, present, *or future* time [19]. If the business time is in the future, the event describes a future (planned or anticipated) event. An example of a future event is given in the example in Section 2 where a patient is expected to be discharged at some future time. The discharge event is a future event since it will only be valid in the real world in the future.

Note that the notion of having different, changing business time for an event means that individual events must be represented in the system as first class objects, e.g., "the take-off of flight 871" remains "the take-off of flight 871" (the *same* event) even if the take-off time is shifted shortly before take-off to send the flight back for additional de-icing.

The decision or processing time of an event is important to resolving issues related to event bitemporality. In particular, the decision time in relation to the temporal properties of an event can influence the type of decision to be taken if temporal requirements are to be preserved. The decision time is the time an event is processed or some decision is taken based on an event. It can also be a decision taken about an event when the event is updated within the system. By necessity, the decision time (as provided by the operating system of the BPMS platform) is always the present at the time the event is processed. We therefore denote the decision time within the description of an event or of a business rule defining the reaction to an event as $\mathbb{NOW}$.

Since the business time is mutable, it can be changed when additional information is obtained or errors are detected. In such a situation, the process must adapt to the changed

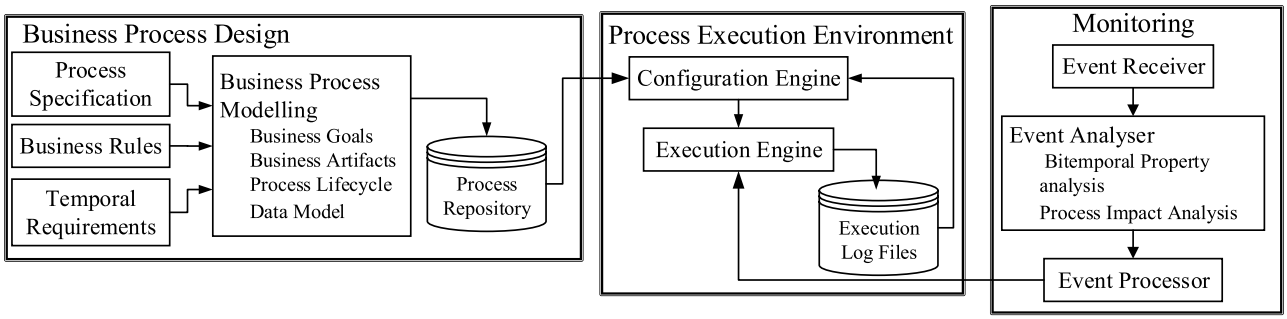

Figure 2: Architecture Overview

time. Changes made to business time may result in an earlier or later business time with respect to the original recorded time. The decision time on the other hand is automatically assumed to be the current time if the event is reprocessed as a result of notification of a change in business time.

DEFINITION 2. *(Change in Event Bitemporal Properties). Given a bitemporal event e with decision time dt and business time bt, a change in bitemporal properties updates dt to the new decision time dt' (defined as the time provided by the platform at the moment of processing the event) and bt to a new business time bt' where, $dt' > dt$, i.e., the new decision time is always greater than the original decision time, but $(bt' > bt) \vee (bt' < bt)$, i.e., the business time can be updated to a new time that is either greater than or less than the original business time.*

This definition provides the allowable changes that can occur to bitemporal properties during runtime. Changes can occur to events that are considered valid in the real world (events up to the present time) or expected to be valid in the future (future events). This paper focuses on future events.

The following holds for changes to future business time (future events): the updated business time will still be a future time if it is greater than the current time, i.e., $(\mathbb{NOW} < bt' < bt) \vee (\mathbb{NOW} < bt' > bt)$. On the other hand, the new business time will not be a future time if it is less than or equal to the current time, i.e., $\mathbb{NOW} \geq bt' < bt$.

A detailed description and classification of bitemporal properties and their relationships in both isolated and distributed systems is given in our earlier work [21].

3. ARCHITECTURE OVERVIEW

In this section, we present a general overview of our architecture which includes a discussion of components associated to a business process management system. The conceptual architecture of our approach, as shown in Figure 2, consists of three main components: business process design, monitoring, and process execution. The business process model is designed in the process design component. Instances of the model are managed in the execution environment which controls the reconfiguration of processes. The monitoring component is activated once execution of the process begins. This stage presents new techniques for bitemporal event analysis which serves as the basis for triggering process reconfiguration. Details of each component in our architecture is discussed in the following subsections.

3.1 Business Process Design

The business process model is designed by this component and must comply to business rules and temporal requirements of the process. Temporal requirements include time constraints specified by a business process designer or constraint by a partner process. Measures must be put in place to ensure that these requirements are not violated. Business rules are also specified at this stage and they must ensure that the process specifications and temporal requirements are not violated. These requirements serve as constraints for the modelled process to ensure compliance of the designed model to the requirements.

EXAMPLE 1. *A sample homecare process is shown in Figure 3. We use the Business Process Model and Notation (BPMN) specification for the sake of simplicity to give an overview of the process. It consists of two interacting process: hospital and homecare processes with the latter being the focus of this paper. The hospital process serves as a partner process requesting services from the homecare process. The latter schedules home care visits by care workers for patients. Scheduling requires assessing the patients medical status which is necessary in determining the required level of skill of the homecare workers to be assigned to the patient and the medical equipment needed. After which homecare workers and the medical equipment are organised simultaneously.*

We adopt time patterns [11] used to describe time constraints in business processes to specify temporal constraints on our process. These temporal constraints include minimum and maximum duration as shown in Figure 3. The temporal constraints are specified in the process model to ensure that activities and, consequently, the entire process are performed within some time limit.

The underlying data model as well as the business artifact lifecycles of the process can easily be determined using the BPMN process. This is essential to perform formal analysis on the process. We shall discuss this further in Section 4.

3.2 Process Execution Environment

The process execution environment has two main functions: (1) The execution of the processes is controlled in the execution engine which manages all process instances as well as enforcing all process constraints. The history of instances are stored in the execution log files which can be utilised by the (2) configuration engine that configures processes to dynamically design parts of a process. In addition,

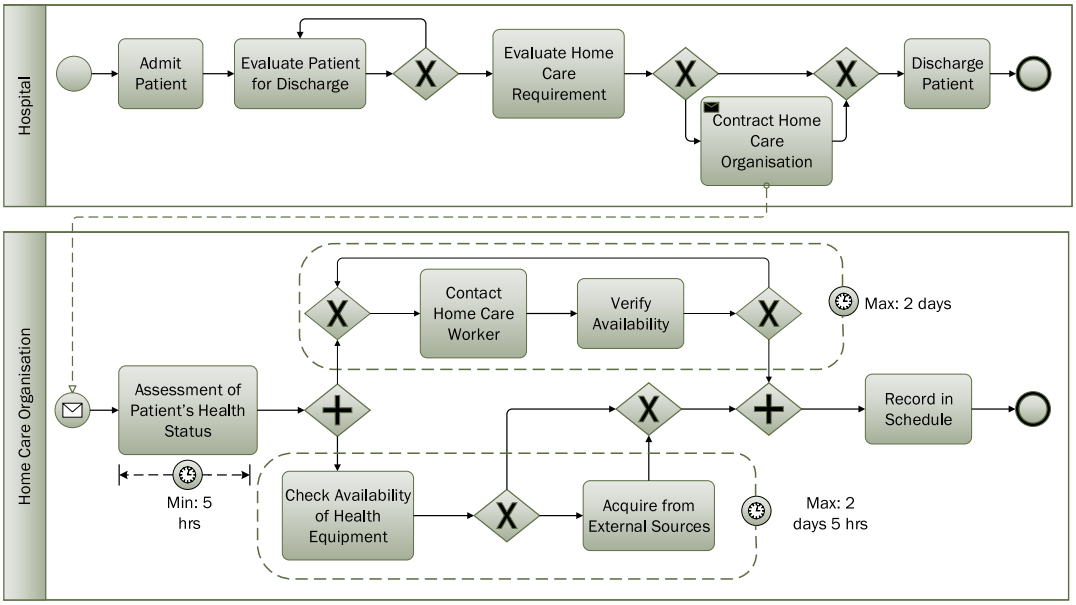

Figure 3: Homecare Process

it reconfigures processes when requirements are changed or execution constraints are violated.

Reconfiguration is essential when the existing execution path of the process instance may result in violation of (temporal) constraints. We determine potential violations by analysing the bitemporal properties of events and using this information to specify constraints for the configurator, possibly triggering reconfiguration. Based on the temporal information available and the constraints of the entire process, the scope of reconfiguration can be determined.

3.3 Monitoring

The monitoring component forms the core part of our work. Its main function include determining the impact of changes in bitemporal properties of events at runtime. This information can be used by the configuration engine to manage contingencies by reconfiguring the process. This component consists of an event receiver, analyser and processor.

Events sent by external systems or produced within the business process are captured in the event receiver. The events are then passed to the event analyser which queries bitemporal properties of received events and determines the impact of these events on the business process. In this work, we introduce new concepts and techniques for bitemporal property impact analysis in business processes. This is treated in detail in the following section. Information from the event analyser is then sent to the event processor which utilises this information for controlling process execution as well as triggering process reconfiguration.

4. IMPACT ANALYSIS

Determining the scope of impact with regards to changes in bitemporal properties of events on business processes is a first step to ensuring that contingencies are handled in a timely as well as effective manner. Our approach to achieving this consists of four main steps as shown in Figure 4 with the aim of detecting changes in the bitemporal properties of an event and then determining how this impacts the business process. This is achieved by determining the impact of change on the data model of the business process. This step is followed by analysing the impact of change on the lifecycles of the business artifacts. Additional information in the execution log files is used to determine the process history as well as facilitate in trace analysis of the instance. This information will aid in determining when a process instance requires reconfiguration. Each step is discussed in detail in the following subsections.

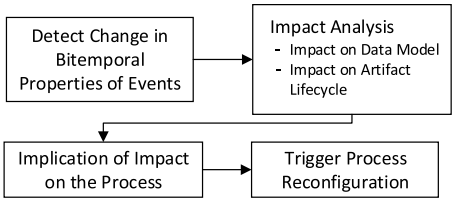

Figure 4: Approach Overview

4.1 Data Model

We first identify the data objects (business artifacts) and events of a business process and then determine relationships that exist between these data objects. We use the terms data object and business artifact interchangeably to represent business objects with lifecycles. This information is used to design a data model for the business process describing the various data objects of the process model and their relationships. The data model is essential in determining how changes occurring in a particular object can impact other object. In our case, we separate the business artifacts from the events. The artifacts form the domain model and the events form the event model. The relationship between both models is identified as well. The data model for our homecare example is shown in Figure 5.

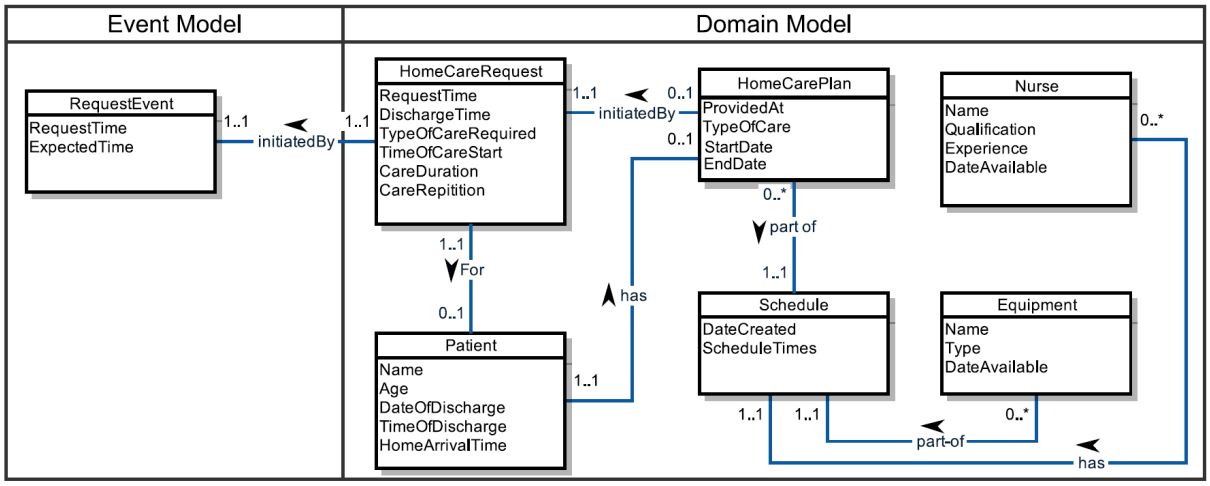

Figure 5: Homecare Data Model

EXAMPLE 2. *The data model, as shown in Figure 5, is based on the homecare example. It consists of two aspects: the domain model and the event model. The latter consists of the request event which initiates the entire process. This event has bitemporal properties (with the request time being the decision time and expected time being the business time) as its attributes. The domain model consists of all business artifacts in the homecare process model that are relevant to providing home care for a patient. The domain model in Figure 5 consists of the home care request, home care plan, nurse, patient, schedule, and equipment as main business artifacts.*

These instances have attributes that may be updated as additional information is received. An event is an instance of the event model and its attributes may be updated during execution. This change may affect artifacts of the business process, resulting in their attributes being changed accordingly. Therefore, changes in bitemporal properties of an event occurring in a business process must be taken into account during both design and runtime to manage contingencies. To achieve this, we need to determine which attributes of an artifact will be affected by these changes.

Identifying changes to bitemporal properties during execution is the first step in determining how the change affects business objects at runtime. We achieve this by determining all temporal constraints associated with the process and the relationships between temporal attributes of business artifacts in the data model. This information aids us in finding dependencies between bitemporal properties of the event model and temporal attributes of artifacts in the domain model. Both direct and indirect relationship between attributes of the domain model and the event model can be determined in a transitive manner.

An informal description of this process is as follows. All entities that have a direct relationship with the event instance are identified. If these entities have time components as part of their attributes, then they may possibly be affected by the change in bitemporal properties of the event instance. Further, entities that have indirect relationships to the event model are identified. These include entities that have relationships to entities with direct relationships with the event instance. Additionally, entities that have relationships with indirect entities are considered as well. Hence, a large number of entities may be affected by the change in bitemporal properties of an event instance.

To be more precise, we use a constraint language to specify dependencies between temporal properties in the data model as shown in Listing 1. Object Constraint Language (OCL) [13] is used to represent these constraints from the data model. Using these constraints, we can identify both direct and indirect dependencies between bitemporal properties of events and artifact attributes. For instance, artifact home care request's attribute time of care start has a direct dependency with event attribute expected time (business time) whereas discharge time has an indirect dependency as it is linked to time of care start which is in turn linked to expected time. The start date attribute of the Home care plan artifact is dependent on time of care start attribute of the request artifact, consequently establishing an indirect dependency between start date and expected time.

For a longer example of dependency, consider the following: the date available attributes of artifacts nurse and equipment are dependent on the schedule times attribute of the schedule artifact, which is dependent on the start time attribute of the home care plan artifact, which in turn is dependent on the time of home care start attribute of the home care request artifact, which is dependent on the expected time attribute of the request event. Thus, a dependency between date available and expected time can be determined.

Using this approach, all dependencies between bitemporal properties of events and temporal attributes of artifacts in the data model can be determined. Moreover, possible violations of temporal constraints as a result of change in bitemporal properties can be determine by using the determined temporal dependencies. Possible threshold and extent of changes on the affected temporal attributes is discussed in Section 4.3.

4.2 Artifact Lifecycle

We now consider how the changes to bitemporal properties of events affect the lifecycle of the business artifacts of the process. In other words, we need to study the behaviour of the business artifacts and how changes in bitem-

Listing 1: Temproral Constraint Representation using OCL

```
Context: HomeCareRequest
Inv: self.dischargeTime < self.timeOfCareStart
Inv: self.timeOfCareStart = initiatedBy.expectedTime

Context: Patient
Inv: self.homeArrivalTime < self.for.initiatedBy.expectedTime
Inv: self.homeArrivalTime > self.for.dischargeTime

Context: HomeCarePlan
Inv: self.startDate = initiatedBy.timeOfCareStart

Context: Schedule
Inv: self.dateCreated >= partOf.initiatedBy.requestTime
Inv: StartScheduleDate self.scheduleTimes >= partOf.startDate
Inv: EndScheduleDate self.scheduleTimes <= partOf.endDate

Context: Nurse
Inv: self.dateAvailable = partOf.scheduleTimes

Context: Equipment
Inv: self.dateAvailable <= partOf.scheduleTimes
```

poral properties affect this behaviour. In this subsection, we shall discuss the formal notation used in modelling artifact lifecycle and methods used in determining the impact on event bitemporal change on the artifact lifecycles.

4.2.1 Modelling Artifact Lifecycle

Behaviour diagrams [18, 9], based on Petri nets [16], are used to graphically represent artifact lifecycles of business processes. This approach ensures the modelling of possible lifecycles of business artifact instances by states, activities, and arcs corresponding to Petri nets' place, transitions and, arcs [9]. The definition of artifact lifecycle as presented in [18] is given in Definition 3.

Definition 3 (Artifact Lifecycle). *An artifact lifecycle $B_O = (S_O, T_O, F_O)$ of an artifact type O (the subscripts are omitted if O is understood) consists of a set of states $S \neq \emptyset$, a set of activities $T \neq \emptyset$, $T \cap S = \emptyset$, and a set of arcs $F \subseteq (S \times T) \cup (T \times S)$, such that $\forall t \in T : (\exists s \in S : (s,t) \in F) \wedge (\exists s \in S : (t,s) \in F)$ and $\forall s \in S : (\exists t \in T : (s,t) \in F) \vee (\exists t \in T : (t,s) \in F)$. There is a distinguished state in S, the* initial state α*, where for no $t \in T : (t,\alpha) \in F$; and α is the only state with this property. There is a nonempty set of distinguished states of S, the* final states Ω*, where for no $s \in \Omega$ and no $t \in T : (s,t) \in F$ and the states in Ω are the only states with this property.*

Similar to Petri nets, behaviour diagrams can be used to determine the legal sequence of activities and states. In addition, instances of objects can be represented by tokens, similar to Petri nets. However, unlike Petri nets where a transition is automatically fired if every prestate contains a token, an activity in a behavior diagram must be explicitly invoked for an object which is in every prestate of the activity. In addition, and unlike Petri nets, activities take time. Therefore, during the execution of an activity on an object, the object resides in an implicit state named after the activity. This state is referred to as an activity state. Thus, we can say that every instance of an object type is at any point in time in one or several (activity) states of its object type, which are jointly referred to as lifecycle state [18].

Definition 4. *(Lifecycle State). A lifecycle state (LCS) σ of an Artifact type O is a subset of $S \cup T$. We denote the initial LCS α by A.*

Definition 5. *(Start and Completion of an Activity). An activity $t \in T$ can be started on a lifecycle state σ, if the set of prestates of t is contained in σ. The start of activity t on LCS σ yields the lifecycle state $\sigma' = \sigma \backslash \{s \in S | (s,t) \in F\} \cup \{t\}$. An activity $t \in T$ can be completed on a lifecycle state σ, if t is in σ. The completion of activity t on σ yields the lifecycle state $\sigma' = \sigma \backslash \{t\} \cup \{s \in S | (t,s) \in F\}$.*

Example 3. *The homecare example is composed of six main business artifacts. These artifacts constitute the domain model in the data model shown in Figure 5. The lifecycle of each artifact based on Definition 3 is shown in Figure 6. States are represented by circles while activities are represented by rectangles. Relationships are represented by arcs connecting states to actions and actions to states. Only the activities in each artifact are labelled in the figure. Intuitively, each pre- and post-state of an activity can easily be determined. The home care request artifact lifecycle include the activities;* create *request,* send *request,* receive *request,* assess *request, and* reject *request or* confirm *request. The pre and post-state of activity* send *request are request* created *and request* sent *respectively.*

Note that an activity that is not labelled does not have any effect on the artifact but is required to preserve the syntax of behaviour diagram notation [3].

4.2.2 Trace Analysis of Instances

The legal sequence all lifecycle states that an instance of each artifact will take during execution is established in the artifact lifecycle. This is important in ensuring that at each point in time during the runtime of the business process, three important factors can be determined about an instance of an artifact, i.e.:

1. Its current lifecycle state;
2. All states that have already occurred in the lifecycle of the process (lifecycle occurrence);

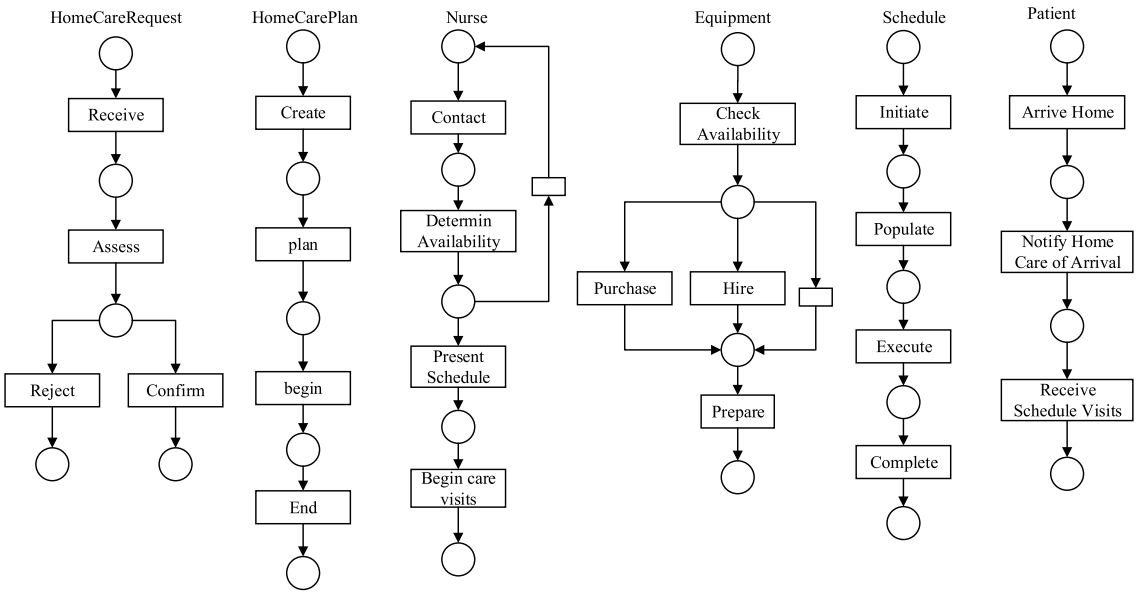

Figure 6: Lifecycle Model

3. All states that have not occurred yet but can possibly occur (reachable lifecycle states).

A particular sequence of lifecycle states of an artifact type is referred to as life cycle occurrence of that artifact type [9].

Definition 6. *(Lifecycle Occurrence). A lifecycle occurrence (LCO) γ of object type O is a sequence of lifecycle states $\sigma_1, ..., \sigma_n$, such that $\sigma_1 = A$, and for $i = 1, ..., n-1$ either $\sigma_i = \sigma_{i+1}$, or there exists an activity $t \in T$ such that either t can be started on σ_i and the start of t yields σ_{i+1} or σ_i contains t and the completion of t yields σ_{i+1}. Any subsequence of γ is called a partial LCO. An LCOγ is called complete, if $\sigma \subseteq \Omega$.*

Definition 7. *(Reachable Lifecycle States). The set of lifecycle states reachable from LCS σ, written $R(\sigma)$, contains every LCS σ' of O that can be reached from σ by starting or completing any sequence of activities in T.*

The partial LCO of an artifact instance is essential in determining which lifecycle states of the instance have occurred. In addition, this information is useful in determining LCS that need to be revisited due to change in temporal properties of the data model. In our approach, we determine the lifecycle state that an artifact instance is in when a change occurs in bitemporal properties of an event resulting in a change in temporal attributes of the artifact instance. We denote this lifecycle state as σ_b such that $\sigma_1 \leq \sigma_b \leq \sigma_n$, where σ_1 is the initial LCS and σ_n is the finial LCS of the artifact instance. With this information, we can determine the σ_b of an artifact instance, and the sequence of lifecycles completed before σ_b, i.e., $\sigma_1, ..., \sigma_b$. In other words, we can find the partial LCO of an artifact instance up to σ_b.

Example 4. *From the artifact lifecycle models in Figure 6 of the homecare process, we determine the σ_b, and the partial LCO of the instances of each artifact during runtime. This is presented in Table 1. From the table, we can see that home care request and equipment artifact instances are complete such that, their $\sigma_b \equiv \sigma_n$, i.e.,* confirmed *and* prepared *respectively. As such, their lifecycle occurrences is not partial but complete. On the other hand, the patient artifact has not begun its lifecycle and, hence, does not have a LCO. All other artifact instances have partial lifecycle occurrences with their $\sigma_1 < \sigma_b < \sigma_n$.*

Knowing the σ_b of an artifact instance alone is not enough in determining which LCS within the LCO has been affected by the change in temporal attribute of the artifact. We need to determine LCSs of the artifact instance that are dependent on temporal attributes of the artifact. These are termed the *critical lifecycle states* and can potentially be affected by changes in bitemporal properties of events.

Definition 8 (Critical Lifecycle States). *A critical lifecycle state σ_c is a LCS of an artifact instance that is time dependent, i.e., an activity t associated with σ_c is expected to start and/or complete at a particular time.*

Intuitively, all LCSs that are expected to start or complete at a particular time are time dependent. These LCSs fulfil certain temporal attributes or constraints. For example, the activity begin care plan in the critical LCS begin must start on start date. The critical lifecycle states of artifacts in our example are shown in Table 2. For convenience, we use the sequence start and completion of activities within the critical lifecycle states of the artifact. We use $s(t)$ and $c(t)$ to represent the start and completion of activity t respectively. A corresponding temporal attribute in the data model (Figure 5) is added to each start or completion of activity in Table 2 to demonstrate dependencies. Therefore, all critical LCSs are associated with the temporal constraints in Listing 1.

Critical lifecycle states can be determined during design time by identifying all time dependent states. The critical lifecycle states of an artifact instance is essential in facilitating process reconfiguration towards contingency management. As activities in these states are time dependent, their

Table 1: LCOs of Artifact Instance

Artifact	σ_b	Partial LCO
HomeCareRequest	confirmed	create, created, send, sent, receive, received, assess, assessed, confirm
HomeCarePlan	planned	create, created, plan
Nurse	Schedule Presented	contact, contacted, check availability, availability checked, present schedule
Equipment	prepared	check availability, availability checked, hire, hired, prepare
Schedule	populated	initiate, initiated, populate
Patient	–	–

Table 2: Activities in Critical LCS and Their Respective Temporal Attributes

Artifact	Activities in Critical LCS and Temporal attributes
HomeCareRequest	
HomeCarePlan	s(plan) *at* HomeCareRequest.RequestTime, s(begin) *at* HomeCarePlan.StartDate, c(begin) *at* HomeCarePlan.EndDate
Nurse	c(availability checked) *at* HomeCarePlan.StartDate, c(present schedule) *at* Schedule.DateCreated, s(begin care visits) *at* HomeCarePlan.StartDate
Equipment	s(prepare) *at* HomeCareRequest.RequestTime
Schedule	s(execute) *at* HomeCarePlan.StartTime
Patient	s(arrive home) *at* HomeCareRequest.DischargeTime

execution must be altered if a possible violation of temporal constraints is detected. Thus, managing contingencies by adapting the process through process reconfiguration.

4.3 Implications

So far, we have discussed how change in bitemporal properties of events can affect artifact attributes. In addition, the critical lifecycle states of each artifact instance in its lifecycle occurrence can be determined. However, these two steps have been done in isolation and must, therefore, be combined to determine the full effect of changes to bitemporal properties of events and what kind of contingency plan needs to be executed to resolve the effect. This is achieved in two steps: determining the significant effect of change in bitemporal properties on artifact attributes, and determining the extent of reconfiguration required.

We first determine the magnitude of change of the bitemporal properties of an event. This is essential to determine if the change has any significant effect on attributes of an artifact and the entire process as a whole. For instance, a 'small' change in business time may not affect artifact attributes much or the entire process as a whole. However, a 'big' change may have a large impact on the process and will, therefore, require process reconfiguration to resolve the effects. Optional threshold values that can be specified at design time for the process may be used to determine when reconfiguration is required. These threshold values, if specified, can be captured in the temporal constraints of the process, providing allowable extension of the process.

Definition 9 (Magnitude of change). *Given bt and bt' as the original and updated business time of an event (from Definition 2), the magnitude of change Δ is given as the difference between bt and bt', i.e., $\Delta = bt' - bt$.*

When Δ is a negative value, then the business time is updated to an earlier time with respect to its original time. On the other hand, Δ is positive if the new business time is set at a later time than the original time.

Definition 10 (Threshold Values). *A threshold value for a process is specified to indicate the allowable change that can be permitted without the need for reconfiguration, i.e., minimum threshold* Threshold_{min}*, or that exceeds the usefulness of the reconfigurator, i.e.,* Threshold_{max}*.*

Note that thresholds can be specified in both direction, i.e., for both reduced and increased business time values. Therefore, it can be specified for both positive and negative values of Δ. To determine the significance of change in bitemporal properties of events, we compare the magnitude of change with the threshold to determine the relationship that exists between them. Δ only has an impact on an artifact if it is greater than the set minimum threshold. Therefore, the effective part of Δ which can trigger process reconfiguration needs to be calculated.

We assume that the magnitude of change lies between the minimum and maximum thresholds, then the *Effective Magnitude of Change* Δ_{Eff}, i.e., the magnitude that will actually affect the process, is given as: $\Delta_{\text{Eff}} = \Delta - \text{Threshold}_{min}$. Note that if thresholds are not specified at design time, then $\Delta = \Delta_{\text{Eff}}$. Just as Δ, Δ_{Eff} can be negative or positive depending on the relationship of the updated business time with the original business time. Δ_{Eff} is applied to each attribute of artifact instances that are affected by the change in bitemporal properties of the event.

In Section 4.1, we determined which attributes of artifact instances will be affected by a change in bitemporal properties of an event. Now we can determine the extent of change with Δ_{Eff}. This value effectively increases or decreases the values of these temporal attributes, consequently, changing the tendencies of violation of temporal constraints. The temporal attributes affected are those attributes established in Section 4.1 by using direct and indirect temporal dependencies. This change thus affects the activities performed in the artifact lifecycle, possibly triggering reconfiguration.

After determining how much the temporal attributes of artifacts have been affected by a change in bitemporal properties of an event, the next step is to determine how the change in values of temporal attributes affects the activities and how this leads to process reconfiguration. However, the latter is not our focus but we deal with facilitating reconfiguration by specifying additional constraints to the already existing constraints of the configurator.

Each temporal attribute that has been altered has a corresponding (critical) lifecycle state(s) within the artifact lifecycle. Given that this critical state is not in the current partial lifecycle occurrence of the artifact, i.e., $\sigma_c > \sigma_b$, then its timing can be altered along with all other temporal states associated with it. On the other hand, if the critical lifecycle state is in the partial lifecycle occurrence of the artifact, i.e., $\sigma_c \leq \sigma_b$, then this lifecycle state may be revisited and preceding states altered to manage the contingency. The naive approach will be to set up event-condition-action (ECA) rules [14] to trigger reconfiguration so as to alter the entire execution time of the process; such that the execution time is either maintained, increased, or reduced based on Δ_{Eff}.

However, these types of constraints do not provide enough information for the configurator to determine the required type of reconfiguration. Thus, all artifact attributes affected by the change as well as critical artifact lifecycles must be considered. For instance, in our example, the resulting Δ_{Eff} from business time updates has to be catered. Assuming the business time is changed to an earlier time, then, the patients care plan as well as schedule must be extended to cover the Δ_{Eff} period. Moreover, nurses and required equipment must be made available within this period as well.

Our approach to setting up additional constraints for the process configurator is by altering the temporal attributes of artifacts in the data model as well as the critical LCS. The existing temporal attributes of business artifacts that have some dependency on the bitemporal properties of events are augmented by the effective magnitude of change. For example, as shown in Listing 2, the time of care start and start date attributes of the home care request and home care plan artifacts respectively are being updated. The updated attributes introduce constraints into the process. These constraints may be newly introduced or override some default constraints for the particular process instance.

Listing 2: Update Constraints

```
HomeCareRequest.timeOfCareStart =
  HomeCareRequest.timeOfCareStart − ΔEff
HomeCarePlan.startDate =
  HomeCarePlan.startDate − ΔEff
```

The schedule times as well as the availability of the nurses and equipment for the patient must be between the new start and end date. However, since we already planned for the original care period, we only need to plan for the additional period Δ_{Eff}. Therefore, activities associated with critical LCS that are in the partial LCO must be repeated only for the extra period.

EXAMPLE 5. *In our homecare example, we maintain the partial LCOs of the artifact lifecycles in Table 1. From this table, we can determine* $\sigma_1, ..., \sigma_b$. *Further we can determine if a critical LCS is in the partial LCO, i.e.,* $\sigma_c \in \{\sigma_1, ..., \sigma_b\}$. *The activities associated with these LCS need to be activated but for a short temporal constraints. A critical LCS for the* nurse *artifact which is in its partial LCO is* check availability. *This lifecycle state must be activated again, however, not for the entire care planning period but for* Δ_{Eff} *period. In addition, the initial constraint for checking for available nurses may be replaced so as to be able to achieve this in a shorter time interval, i.e., between* $\mathbb{NOW}$ *and* bt'. *Such a constraint for the reconfigurator can be given as follows:* $(duration(checkavailability) < (bt' - \mathbb{NOW})) \wedge (Nurse.dateAvailable \in \Delta_{\text{Eff}})$. *Same applies to the equipment required for assisting the nurse when providing specific services to a home care patient.*

As we have seen so far, the knowledge of the impact of bitemporal properties on the underlying data model of a business process as well as its lifecycle is vastly beneficial to managing contingencies. In addition, possible violations of temporal constraints can be determined in time and avoided. This impact analysis can be extended to areas such as process compliance and performance monitoring within a single organisation as well as in an inter-organisational setting.

5. RELATED WORK

Time management in BPMS is essential for business process optimization with contingency handling capabilities. We discuss approaches to handling time in stages of business process, from designing the process, to process execution and process monitoring:

- **Business process design:** There are a number of approaches for incorporating time in business process models. Ten time patterns have been proposed by Lanz et al [11], which includes commonly used patterns such as activity/process duration and time lags between activities/events. These patterns are adopted in our work to illustrate the temporal requirements of our process (see Section 3.1). Eder et al. [5] presented an approach to represent time constraints in workflows. In their approach, they present an activity model of a timed workflow graph with three temporal attributes: the specified activity duration, the earliest finish time, and the latest finish time of the activity. These are calculated and specified as constraint during design time to ensure that activities and, hence, the entire process is performed within some specified time. Only single time dimensions are considered in all these approaches.

- **Process Execution:** During execution, time constraints of the process must be preserved. This has led to development of approaches for the preservation of temporal requirements of processes during process execution. Some of these works specify internal activity deadlines to ensure that the overall process deadline is not violated [4]. Other approaches provide flexibility solutions in order to adapt a process that may potentially violate some temporal constraints. Pichler et al. [17] presented five intervention strategies (optional execution, parallelization of sequence, alternative path, dynamic service selection, early termination) during runtime as a means of meeting process deadlines. Concepts introduced in these works are adopted in the process reconfiguration. These do not consider two dimensions of time in their approaches as well.

- **Process Monitoring:** Some work outside the domain of BPM considers detecting and analysing bitemporal events. Artikis et al. [2] presented a two time dimensional approach for dealing with uncertainties in events; where, the detection time of the event (decision time) is known, but its exact occurrence time (business) is not known. A similar case is discussed in [1], where the business time of a transaction in a bitemporal database is indeterminate. Our approach does not focus on uncertainty and only considers events with known bitemporal properties.

  Furche et al [7] investigated bitemporal complex event processing of web events. They distinguished between event occurrence time and detection time and proposed an event processing language that can be mapped to standard SQL. The main difference to this work is that they did not focus on business processes and how bitemporal events affect process execution.

Bitemporal events have not been the focus of temporal BPM approaches, thus its impact and implication in this area has not been tackled. The difference between business and decision time as well as their updates and histories are not considered in existing works. Approaches that consider dimensionality of time are outside the domain of BPM and do not consider how it impacts business processes. Hence, there are no existing concepts and techniques for handling bitemporal events in business processes. Those provided in this work can be used to improve existing approaches to contingency management while laying the foundation for more research in this area.

6. CONCLUSION

In this paper, we deal with bitemporal properties of events occurring in business processes and how a change in these properties affects the data model and the lifecycle of the process. We discover that a change in bitemporal property may impact both the data model and behaviour of the process. In addition, how much the change impacts the process is explored. Our aim is to provide useful information that can, possibly, facilitate process reconfiguration as a means of managing contingencies.

Future work shall incorporate bitemporal property relationships discussed in [21], such as late events. Furthermore, we look forward to incorporating our techniques into the reconfiguration tool developed by Grossmann et al. [8].

7. ACKNOWLEDGMENT

This research was partially funded by the Data to Decisions Cooperative Research Centre (D2D CRC).

8. REFERENCES

[1] L. Anselma, P. Terenziani, and R. Snodgrass. Valid-time indeterminacy in temporal relational databases: Semantics and representations. *IEEE TKDE*, 25(12):2880–2894, Dec 2013.

[2] A. Artikis, O. Etzion, Z. Feldman, and F. Fournier. Event processing under uncertainty. In *Proc. DEBS*, pages 32–43. ACM, 2012.

[3] R. M. Dijkman, M. Dumas, and C. Ouyang. Semantics and analysis of business process models in BPMN. *Information and Software Technology*, 50(12):1281 – 1294, 2008.

[4] J. Eder, E. Panagos, H. Pozewaunig, and M. Rabinovich. Time management in workflow systems. In ., editor, *Proc. BIS*, pages 265–280. Springer, 1999.

[5] J. Eder, E. Panagos, and M. Rabinovich. Time constraints in workflow systems. In *Proc. CAiSE*, volume 1626 of *LNCS*, pages 286–300. 1999.

[6] Y. Engel, O. Etzion, and Z. Feldman. A basic model for proactive event-driven computing. In *Proc. DEBS 2012*, pages 107–118. ACM, 2012.

[7] T. Furche, G. Grasso, M. Huemer, C. Schallhart, and M. Schrefl. Bitemporal complex event processing of web event advertisements. In *Proc. WISE 2013*, volume 8181 of *LNCS*, pages 333–346. 2013.

[8] G. Grossmann, M. Schrefl, and M. Stumptner. A conceptual modeling approach for web service composition supporting service re-configuration. In *Proc. APCCM*, pages 43–52, 2010.

[9] G. Grossmann, M. Schrefl, and M. Stumptner. Design for service compatibility. *Software & Systems Modeling*, 12(3):489–515, 2013.

[10] K. Kulkarni and J.-E. Michels. Temporal features in SQL:2011. *SIGMOD Rec.*, 41(3):34–43, Oct. 2012.

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

[12] M. Nicola. Best practices: Temporal data management with DB2. Technical report, IBM, 2012.

[13] Object Management Group. *Object Constraint Language 2.4*, 2014.

[14] N. W. Paton and O. Díaz. Active database systems. *ACM Comput. Surv.*, 31(1):63–103, Mar. 1999.

[15] O. P. Patri, V. S. Sorathia, A. V. Panangadan, and V. K. Prasanna. The process-oriented event model (poem): A conceptual model for industrial events. In *Proc. DEBS*, pages 154–165. ACM, 2014.

[16] J. L. Peterson. Petri nets. *ACM Comput. Surv.*, 9(3):223–252, Sept. 1977.

[17] H. Pichler, M. Wenger, and J. Eder. Composing time-aware web service orchestrations. In *Proc. CAiSE*, volume 5565 of *LNCS*, pages 349–363. 2009.

[18] M. Schrefl and M. Stumptner. Behavior-consistent specialization of object life cycles. *ACM Trans. Softw. Eng. Methodol.*, 11(1):92–148, Jan. 2002.

[19] R. Snodgrass. Temporal databases. In *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, volume 639 of *LNCS*, pages 22–64. 1992.

[20] R. J. Wieringa. *Design methods for reactive systems: Yourdon, Statemate, and the UML*. Elsevier, 2003.

[21] J. Wondoh, G. Grossmann, D. Gasevic, M. Reichert, M. Schrefl, and M. Stumptner. Bitemporal support for business process contingency management. In *Proc. in ER Workshops*, volume 9382 of *LNCS*, pages 109–118. 2015.