# Contingency Management for Event-Driven Business Processes

John Wondoh(✉), Georg Grossmann, and Markus Stumptner

University of South of Australia, Adelaide, Australia
john.wondoh@mymail.unisa.edu.au, {georg.grossmann,mst}@cs.unisa.edu.au

**Abstract.** In the past two decades, business process research has focused on process flexibility to facilitate the operation of business processes in an open and dynamic environment. This is important to ensure that processes accurately reflect and handle changes occurring in the real-world. While substantial existing work has investigated changes in business processes, the contingency management of running processes did not receive sufficient attention, mainly because events are considered to be immutable. Yet high-level business events have been shown to be subject to changes. To be able to capture such changes, business events have to be considered as bitemporal, where the occurrence (scheduled) time and detection time of events are differentiated. Modifying an event's content may result in a contingency that has to be handled appropriately. For instance, the scheduled time of a planned event in a process may change, which has an impact on subsequent events. In this work, we present an approach to capture bitemporal mutable events in business processes, assess the scope of changes and provide an approach for specifying contingency plans.

**Keywords:** Contingency plans · Bitemporal mutable events · Business processes

## 1 Introduction

In today's enterprise, it is crucial for business processes to be capable of operating in an open and dynamic environment. This requirement has resulted in rigorous business process research aimed towards ensuring the flexibility and adaptability of processes during runtime [1,17]. This requires the anticipation and categorisation of possible contingencies that may result from changes in the execution environment.

In *event-driven business process management* (ED-BPM), events are the main citizens in a business process, and are used for monitoring and controlling the business process [3]. Flexibility in ED-BPM is driven mainly by events. The management of contingencies is driven by the occurrence of events associated with each contingency. While substantial existing work has focused on the issue of categorising contingent events to facilitate runtime flexibility, there has been little work towards the handling of modification of business events. This is because


H. Panetto et al. (Eds.): OTM 2017 Conferences, Part I, LNCS 10573, pp. 314–333, 2017.
https://doi.org/10.1007/978-3-319-69462-7_21

traditionally, events are considered to be immutable. Yet high-level business events have been shown to be subject to changes [7,10,20,25]. Since business processes are incapable of detecting such changes in events, there are no existing approaches for handling such changes. Our work focuses on providing techniques for handling contingencies associated with immutability of events in business processes.

Contingency management is an important aspect of business process management (BPM) [21]. It usually follows these steps [9]: (1) identify potential contingency, (2) develop contingency plans, and (3) develop preventive measures. Since contingencies that result from changes in events are not considered in business processes, contingency plans cannot be developed, and preventive plans cannot be implemented. It is crucial to firstly, identify contingencies that may result from changes in events, and secondly, provide plans to handle them. The focus of this paper is to identify contingencies resulting from event modification and synthesising plans to handle them.

Modification of events is a phenomenon that has been investigated in various research areas. For instance, in areas such as web and database monitoring, existing research has investigated the detection and effect of changes in an event [7,10,20]. The previous information of an event, i.e., its old state and the new information of the event, i.e., its new state can still be assessed. To be able to capture changes to events, we need to consider the occurrence and detection times of the event states, i.e., the detection time of the new state will always be greater than the detection time of the previous state. We use the term *bitemporal events* to capture the dual temporal properties of events. Bitemporality of events is necessary to capture the ordering of old and new event information.

This work is aimed at anticipating changes in events, and managing contingency that may result from such changes. The main contributions of our work are as follows:

- We introduce an approach for detecting changes in events, and analysing the scope of the impact of such changes to the business process.
- We provide an approach for designing contingency plans aimed at handling contingencies resulting from event mutation.
- We provide an approach for specifying rules to trigger contingency plans when event contents' are changed.

The remainder of this paper is organised as follows: Sect. 2 provides a discussion of the fundamental concepts used in this work and the running example. Section 3 provides the approach overview. Section 4 provides our approach for determining the scope of bitemporal contingencies in business processes, and Sect. 5 provides our approach for contingency planning. We provide a brief discussion of our overarching project in Sect. 6, followed by a review of related work in Sect. 7. We conclude the paper and discuss future work in Sect. 8.

## 2 Background

In this section, we introduce the fundamental concepts in this work, and provide an example for better illustration. The central concepts considered in this work are *event modification* and the *bitemporal* nature of events.

### 2.1 Bitemporal Mutable Events

The properties of an event include an *event name*, an *identification number*, a *timestamp*, and a *payload*. In conventional event processing systems, only a single timestamp of an event is captured to determine when the event occurred. Other temporal dimensions as described in the temporal database literature are not exploited [18]. This work focuses on bitemporal events [7,24]. The latter has associated temporal properties (1) *occurrence (scheduled) time*; the time the event occurs or is supposed to occur, and *detection time*; the time when the event is detected. Bitemporality is necessary for capturing the history of events after modification.

High-level business events such as requests events may be modified during a process lifecycle. Event modification may occur as a result of two factors: (i) new information about an event is obtained [7,20], or (ii) errors are detected within the event's information that requires correction [20]. In both cases, the result is the modification of the event.

Event modification results in a new state of the event. A state of an event is a representation of the attributes of an event at a given point in time. The state of an event changes when the event content is modified. Given the initial state of an event $s_0$ and its new state $s_1$, let $dt_0$ be the detection time of $s_0$ and let the detection time of $s_1$ be $dt_1$. The relations between the detection time is given as follows: $dt_0 < dt_1$. This relationship is important as it facilitates the ordering of event states. The content of $s_1$ differs from the content of $s_0$. The occurrence time of an event is used to represent when the event occurred. For planned events, i.e., events expected to occur in the future, the actual occurrence time is in the future. In this work, we use the term *scheduled time* instead of occurrence time to allow for the inclusion of planned events.

We considered two types of modifications in this paper: (1) modifying the content of an event, and (2) modifying the scheduled time of an event. The content of an event is modified by replacing the value of an attribute of the event with a new one, deleting an attribute, or adding new attributes to the event. These modifications are demonstrated in the example below.

### 2.2 The Homecare Example

Homecare organisations provide home care services to clients with physical disabilities, clients who require carers during their health recovery process, and the elderly. The homecare organisation provides these clients (patients)[1] with carers

[1] Client is a preferred term in homecare organisation while patient is the preferred term in hospitals.

who visit them at home and provide necessary services to them. Initially, the homecare organisation is made known of a possible client and the necessary information required to provide the service to the client. A typical example is when a hospital makes a request for home care for a patient planned to be discharged from the hospital. Based on the planned discharge date and all the necessary information provided by the hospital, the homecare organisation plans the care schedule for its clients. Once the patient is discharged, homecare services will be provided to the client according to the planned schedule. Constant changes are likely to occur before, during, or after the planning or service provision stage.

The processes in the homecare organisation consist of events, activities, gateways. Some of these events are high-level events such as message events, while others are primitive events signifying the initiation or termination of activity instances. We selected four important processes in the homecare organisation for illustrating potential event changes in the process. The selected processes are the planner, dispatcher, client, and carers process. These processes, modelled in *business process model and notation* (BPMN), are shown in Fig. 1.

**Planner.** The planner process in Fig. 1 is responsible for planning home care services for each client. After receiving the home care request from the hospital, the homecare planner process first assesses the needs of the patient. Based on this assessment, the carers with the required skill level are selected. Firstly, an available carer is selected and his/her skill level is determined. If the skill level is not within what is required for the client, a new available carer is selected. Else the carer is contacted to verify his/her availability for the shifts intended to be allocated to him/her. Once the availability of the carer has been verified, it is recorded in the schedule (homecare plan database). If a carer is unavailable to cover his/her shifts, the database is updated with the carer's availability status for later use. This search is repeated until the plan for each client is completed.

The information required to plan the home care for the client includes the client's name, health status, care type requirement, planned discharge date, and care intensity and duration for the client. This information is sent to the homecare organisation as the request event in the planner process in Fig. 1. In the real-world, this event may undergo changes such that the hospital updates the message sent to the homecare organisation. The planned discharge date may be moved to an earlier or later date, the care duration or intensity may change, and/or the care type may change as well. Simply, the scheduled time of the event, as well as its content may be modified. In extreme cases, the entire request may no longer be necessary and the entire message event cancelled. Another event that may change is the response from carer after they have been contacted to determine their availability, i.e., the response event in Fig. 1. The initial response may change from 'no' to 'yes', or from 'yes' to 'no'. Processes should be capable of handling such resulting contingencies.

**Dispatcher.** The dispatcher process in Fig. 1 is responsible for notifying carers of the next client they need to attend to according to their verified schedule.

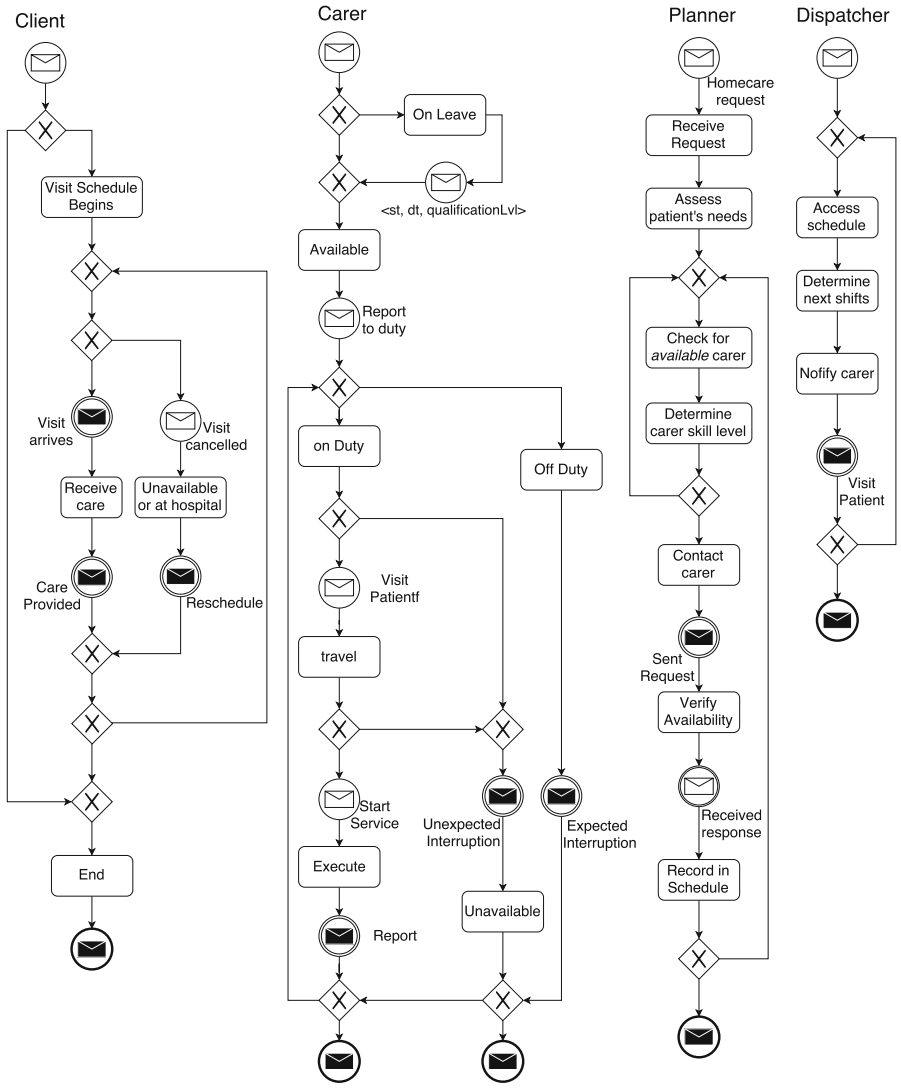



**Fig. 1.** Homecare organisational processes

It does this by accessing the carer's schedule, determining the next shift and notifying them. Usually, the notification should take into consideration the distance the carer has to travel to get to the home of the client. Changes in the dispatcher process are external and may affect the carer and/or client processes.

**Client.** The client process in Fig. 1 shows the care services provided to the client after being discharged from the hospital. The client receives care visits from the carers where home care services are provided to him/her. Occasionally,

home care services cannot be provided because the client may be re-admitted to the hospital, or become unavailable due to personal reasons. The business event likely to change in the client process in Fig. 1 is the visit cancelled event which represents the cancellation of an initiated visit. This event may be dismissed (cancelled) if new information is obtained regarding the return to the client to his/her home. Processes should be capable of handling event cancellations.

**Carer.** The carer is responsible for providing all the necessary home care services to the patient. The process for each carer is given in Fig. 1. A carer may be on leave, but upon returning should notify the organisation of any changes in their physical condition that may impede their work output. For instance, a carer with a high level of qualification may be required to do less work if they develop a back injury. Once the availability of a carer is determined, they are required to report to duty. A carer may be off duty if they have completed their work for the day, or on duty for their shifts. Off duty carers will notify the homecare organisations that they have complete all client services for the day. The on duty carer receives a notification from the dispatcher about the next client to visit. The carer travels to the client's home to provide the services. The carer may not be able to reach the client's home due to some *unexpected interruption*. For instance, the carer's car may breakdown. The carer will notify the homecare organisation about how long he/she may not be available.

Some of the changes that can occur in the events in the carer process are the visit patient event may be changed depending on whether the patient is available at home or not. The dispatch process may change the parameters of the visit patient event to delay the visit by say 1 h. The *unexpected interruption* may not be as severe as initially thought and the carer may reduce the number of hours that he/she may be unavailable. These changes, among others, are represented by the changes in events. The business process should be equipped to handle such changes. In the next section, we shall discuss how we handle changes to events in business processes.

## 3 Approach Overview

The overview of our approach is given in Fig. 2. This consists of two main components, i.e., the event modification component, and the contingency management component. The event modification component detects modification of an event's content or its scheduled time and adds the modified event to the modified event set. A modified event may cause other events within the processing system to be modified as well. This is termed as modification propagation [25]. The provenance component captures the current and previous states of all modified events. This aspect of the work was the focus of our initial work [25]. The output of this component is a set of modified events serving as an input to the contingency management component.

The contingency selector is responsible for selecting the right contingency plan for managing a change in an event. The selection can be done automatically by bitemporal business rule engine or by incorporating the assistance of

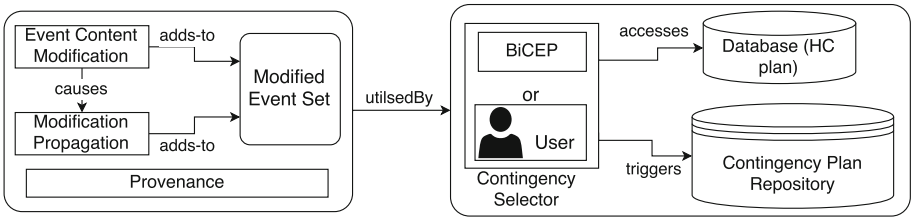



**Fig. 2.** Approach framework

a domain expert. *Bitemporal Complex Event Processing* (BiCEP) engine was designed by [7] purposely for managing updates in bitemporal web events. Our rule engine is based on BiCEP. It takes as input the modified event set, evaluates a set of conditions, and triggers the contingency plan necessary for correcting the deviation. The contingency plans are modelled and stored in process repository and can be accessed by domain experts or the BiCEP engine.

Process models are designed and instantiated during runtime, where activities are executed according to how they are specified. We term this as the passive behaviour of the process. When we trigger contingency plans, the process instance deviates from its design time specification. It no longer passively executes activities, but exhibits a dynamic behaviour. We term this behaviour the *active behaviour* of the process instance. We actively control the behaviour of the process to successfully manage contingencies.

## 4 Contingency Scope

Our contingency plans are triggered by bitemporal rules. These rules, inspired by [7] are a variant of the *event-condition-action* (ECA) rules, where modified events can be detected and processed. When we detect a modification of an event's content, evaluate a set of conditions associated with the rule specific to the event, and then perform a set of actions if the conditions are satisfied. The condition part of a bitemporal rule evaluates the following:

- Event Type: Rules are specified for event types and are applicable to their associated event instances. Once an event instance is modified, its associated rule is activated.
- Nature of deviation: The nature of deviation ($NoD$) resulting from event modification can be put into two main categories:
  - Nature of Scheduled Time Deviation ($NoD_{st}$): This nature of deviation results from modifying the scheduled time of an event. $NoD_{st}$ is a tuple $(\delta, \alpha)$, where $\delta$ is the modification type of the scheduled time modification, and $\alpha$ is a the modification scope of impact on the process.
  - Nature of Event Content Deviation ($NoD_{ec}$): The $NoD_{ec}$ resulting from event modification deals with changes in the content of an event. $NoD_{ec}$ is a tuple $(a, a', \theta)$ where $a$ is an attribute of the event in its initial state,

$a'$ is the attribute of the event in its new state, and $\theta$ is the significant scope of change between $a'$ and $a$.

– Additional Conditions: These are additional conditions required for triggering contingency plans and are specified by domain experts. These conditions may be omitted during contingency planning, however once specified, they must be taken into account when evaluating the rule's conditions.

We shall now proceed to discuss how the nature of deviation can be determined in the following sections.

### 4.1 Modification Type

We ascertain the modification type by comparing the previous and new scheduled times of the modified events. We denote the previous scheduled time as $st$ and the new scheduled time as $st'$. Corresponding to the scheduled times are the detection times of the events. The detection time of the previous event state is denoted as $dt$ and the detection time for the new event state is denoted as $dt'$. We denote the current wall clock time as $\mathbb{NOW}$. The latter corresponds to the current real world time.

Based on the temporal properties associated with events, there exist various relationships between $st$ and $st'$. Such relationships, as provided in [7,10], are illustrated in Fig. 3. The relationships are put into three main categories, namely announcement, modification, and cancellation. The announcement category deals with only one scheduled time; there is no need to draw a distinction. In this category, the scheduled time of the event is not modified. In the second category, the scheduled time of the event has been modified resulting in $st'$. The cancellation category deals with the cancellation of an event, and consequently its scheduled time.

We have excluded the trivial case of no actual change, i.e., when a modification results in the new scheduled time being the same as the previous scheduled time ($st = st'$). Further, since the first category contains no changes, we only introduce it as the initial state of the scheduled time. The two non-trivial categories are the modification and cancellation categories. These categories may potentially result in changes in the running process.

The event announcement category deals with the detection of events, i.e., when the event becomes known in the process. In this category, the focus is on finding the relationship between the detection time and the scheduled time of an event. In Fig. 3, $\mathbb{NOW}$ is the time when the event becomes known in the business process management system (BPMS), and this is compared to the scheduled time of the event. In categories modification and cancellation, the event is already known in the BPMS, however, changes are compared to the current wall clock time that coincides with the time the change occurred.

*Example 1.* From our home care example (see Sect. 2.2), changes in scheduled time of some events will fall under one of the modification types in Fig. 3. For instance, the planned discharge time of the patient from the hospital is a scheduled time. This time is important as it may affect the homecare schedule of

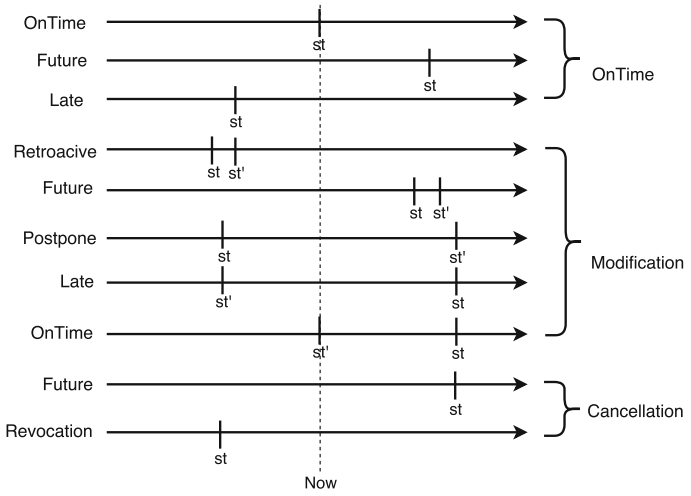



**Fig. 3.** Qualitative temporal comparisons of scheduled time states [7]

the patient. As discussed in the example, the discharge time may be modified. Assume the current time has exceeded the planned discharge time. If the planned discharge time is modified to a later time than the current time, then the modification type for the planned discharge time is future under modification category. Similarly, if the discharge time is modified to a time that is less than the current time, then the modification type for the discharge time is retroactive under modification category.

The modification type provides a qualitative description of the change in the scheduled time of the event. The quantitative effect of this change on the entire process needs to be determined as well. This is addressed in the next section.

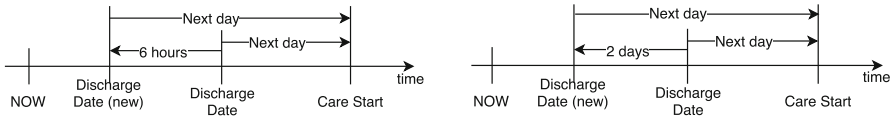
### 4.2 Modification Scope

The modification scope captures the impact of scheduled time modification to process specification. The quantitative temporal distance between the new and previous scheduled times is used to determine to extent of the change. The temporal distance can be calculated as follows:

$$t_d(\delta) = st' - st$$

Where $t_d$ is the temporal distance, and modification type is replaced by the specific type in the modification category. A negative value for $t_d$ represents a change where the scheduled time is modified to an earlier time, i.e., $st' < st$, while a positive value represents a change where the scheduled time is modified to a later time, i.e., $st' > st$.

The temporal distance created during modification may not have an impact on the running process if the change is not significant. To determine whether a




(a) Insignificant Scheduled Time Modification

(b) Significant Scheduled Time Modification

**Fig. 4.** Illustration of scheduled time modification significance

change actually affects process execution, we need to identify if specified threshold values have been exceeded. A *threshold value* is a value specified by a domain expert such that any change that does not exceed this value is disregarded. The threshold value is set for each modification type such that it is only evaluated if the change matches its modification type. We denote the threshold for a modification type as $\textsc{Threshold}_\delta$.

The temporal distance introduced by a modification type, and the threshold value for the modification type can be used to determine the *effective temporal distance* of the change in schedule time, i.e., the scope $\alpha$. The effective temporal distance is the temporal distance that actually affects the process. This is the modification scope $\alpha$ and its value can range from $-\textsc{Threshold}$ to $t_d$, i.e., $-\textsc{Threshold}_\delta \leq \alpha \leq t_d$. The value of $\alpha$ can be determined as follows:

$$\alpha = t_d - \textsc{Threshold}_\delta$$

*Example 2.* In our home care example, let us assume that care services are supposed to begin for the client a day after they have been discharged from the hospital. In Fig. 4, we show two cases where the planned discharge date for the patient has been modified. The new discharge date is 6 h earlier than the original date in Fig. 4a, while the new discharge date is 2 days earlier in Fig. 4b. In both cases, the modification type is future modification as both the new and original discharge times are in the future (i.e., $>\mathbb{NOW}$). Assume care services are supposed to be provided to the patient a day after they have been discharged from the hospital. In the first case, moving the planned discharge date to 6 h earlier may not affect the start date of 1 day after discharge provided the new discharge date is on the same day as the original discharge date. The change is therefore insignificant. In the second case, the new discharge date is two days earlier which makes it impossible to provide services to day patient on the scheduled day. The threshold in this example is 1 day and $t_d$ is 2 days. The modification scope is therefore 1 day. The homecare organisation will have to accommodate the change by adding schedules for the extra day introduced at the beginning of its planned schedule.

### 4.3 Nature of Event Content Deviation

As discussed in Sect. 2.1, event modification may result in the modification of some event attributes. The $NoD_{ec}$ takes into consideration the initial attribute $a$, the new attribute $a'$, and the significance of the change $\theta$. Content modification of an event consists of attribute deletion, attribute addition, and attribute modification. Each of these types of content modification results in a different $NoD_{ec}$ tuple.

- Attribute modification: In attribute modification, the value of $a$ modified such that $value(a) \neq value(a')$.
- Attribute deletion: In this content modification type, an event attribute is deleted from the event content. This is represented by the tuple $(a, -, \theta)$. Attribute $a$ is deleted from the content of the event in the new state and represented by $-$ in the tuple.
- Attribute addition: The addition of an attribute $a'$ to an event content in the new event state is presented by $(-, a', \theta)$. Since there is no previous version of $a'$ in the previous state of the event's content, its non-existence in the previous state is represented by $-$.

If $value(a) = value(a')$, then there was no change in the attribute value and, therefore no significance. This is because not all event attributes in the new state will be modified versions of the old state. In attribute modification, if the modified attribute is not necessary for the execution of the process, then it is not significant. However, if the attribute is necessary for process execution, then the significance of modifying that attribute $a$ is the difference between the $a$ and $a'$. Similarly, if an attribute is necessary for the execution is deleted, then the deletion of the attribute may result in a significant contingency. If a new attribute is added to the content of an event, this content modification is significant if the new attribute is necessary for the execution of the process. Therefore, for attribute deletion and addition, the significance of $NoD_{ec}$ may be either true or false. For attribute modification, $\theta$ is a value representing the difference between $a$ and $a'$.

*Example 3.* In the homecare example, the request event has as part of its attribute *care duration* and *care type*. Assume the care duration was originally 36 days but has been increased to 40 days, then $\theta$ is 4 days. The initial care type may only be to provide support with *activities of daily living* (ADLs)[2]. Extra requirements such as the provision of emotional support may be added to the care type after request event's content modification. The $\theta$ for the care type modification is the extra requirement of emotional supported added to the original attribute value.

[2] ADLs are activities that people do daily without requiring assistance such as eating, bathing, dressing, toileting, walking, and continence. Some clients may require assistance with these activities (see http://www.investopedia.com/terms/a/adl.asp).

## 5 Contingency Planning

Contingency planning is a very important requirement for dealing with the impact of event modification on a business process. Contingency planning for business processes falls under the broad scope of *contingency theory*. The latter holds that business organisations should have the ability to adapt to changes in their contextual environment [6]. Contingency planning has been adopted in BPM for adapting business processes to changes in their execution context [12,14]. It has also been adopted in other areas such as web services composition [4], as well as, operations management research [19].

In this work, contingency plans (CPs) are required to resolve issues that may arise in a business process as a result of scheduled time modification. CPs are modelled during design time, although additional contingency plans can be added later. We now proceed to discuss the design of contingency plans and their application during runtime.

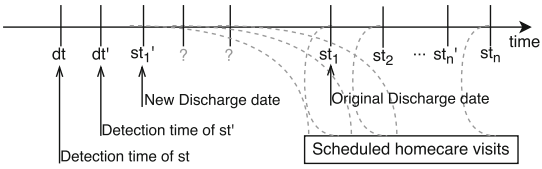
### 5.1 Design Time

At design time, with the assistance of domain experts, contingencies can be anticipated and CPs designed to handle them when they occur during runtime. Synthesising contingency plans should be intuitive, i.e., it should imitate how an actual worker will do it. Ideally, selecting the right contingency plan should be semi-automated. That is, the contingency plan should either be selected automatically using bitemporal ECA rules or manually by a user (see Fig. 2). The manual approach should involve providing all the necessary information regarding available intervention strategies to a domain expert. The latter is required to make the final decision regarding that strategy to consider. During runtime, if a contingency occurs, a contingency plan is automatically selected. The latter may be changed by the user after it has been selected.

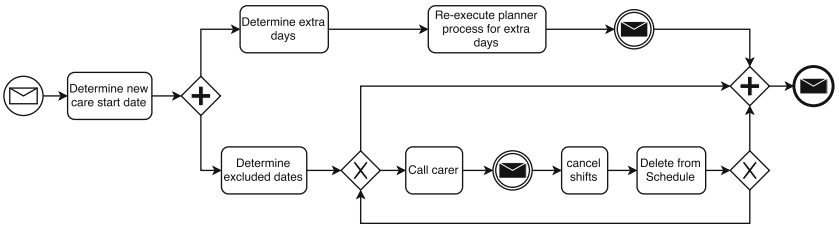
**Modelling Contingency Plans.** CPs are modelled similarly to the process models. The difference between a CP and the process model is that a CP is only triggered when a particular contingency occurs. The CP is similar to a process fragment. The latter is a term used to describe an atomic task, a subprocess, or a sub-graph [23] which replaces a placeholder in a process model based on execution context [1]. Unlike process fragments, CPs do not necessarily replace a placeholder but can be executed simultaneously to the process model. In addition, a CP model has distinctive start and event events.

CPs are stored in a repository which we term as the contingency plan repository (see Fig. 2). A CP becomes active during runtime only if its pre-specified conditions are satisfied. The effect of a CP on the business process can be put into three main categories:

– Reactive Contingency: A CP can be designed to realign a deviating business process with its goal. A carer may be unable to attend to a patient because




(a) Temporal Representation of Discharge Date Modification




(b) Contingency Plan for Handling Discharge Date Modification

**Fig. 5.** Discharge date modification and its corresponding contingency plan

his car broke down. A CP aimed at quickly assisting the carer with a new transportation or getting another carer to take his place is a reactive CP.

- Proactive Contingency: Some CPs are executed to ensure the long term success of the running process. For instance, if the care duration increase from 36 days to 40 days, shifts need to be planned for the extra 4 days. The scheduling of these extra shifts may not be urgent but are required to ensure that the shifts are available when required.
- Compensation Contingency: These are important exceptional handling techniques that can be utilised for contingency planning. We can include activities that compensate changes in requirement. Assume the planned discharge date of the patient is modified to a later date. If carers have already been contacted, we will need to cancel shifts that have been cut off due to the change of date. We can include compensating activities call carers and cancel shifts to resolve the issue for the included time period.

For each type of CP, the process designer needs to identify the right conditions for triggering a particular contingency plan, and the set of actions required to mitigate the effect of the contingency on the process.

A CP process model is essentially a 'small', 'specific' process model with activities, events, and gateways. The start and end events of a cp process model are both message events, i.e., the CP process model required a start event to be initiated and it sends out an end event after it has terminated. The following example describes a cp process model.

*Example 4.* The planner process in Fig. 1 requires a request event with the planned discharge date of the client to initiate planning homecare visits for

the client (see Sect. 2.2). Assume the planned discharge date of the patient is modified to a much earlier date as shown in Fig. 5a. We need to consider the extra days included between the initial discharge date and the new discharge date, i.e., the modification scope. If the care duration is not modified, then some shifts will be unnecessary for the client. From Fig. 5a, the original schedule for the patient ends after scheduled visit $s_n$. Modifying the discharge to an earlier date without modifying the care duration will result in the last scheduled visit being $s'_n$. The shifts planned between $s'_n$ and $s_n$ are unnecessary and must be terminated. The CP is designed to handle these changes. Once the CP has been initialised, the new discharge date is determined, after which two simultaneous paths are executed. The first path deals with the additional shift requirement by determining the number of extra days (scope) and then executing the planner process from Fig. 1 for those days. The second path deals with cancelling shifts for the excluded days. The carers are informed that their shifts for those days have been cancelled, after which the shifts are deleted from the schedule.

**Specifying Bitemporal Contingency Rules.** CPs form the action part of bitemporal ECA rules for business processes. When designing CPs, the conditions for activating a plan is known beforehand. That is, we need to identify possible contingency and consequently design plans to handle them. After designing the CP, we need to include the rule for activating the CP in our system. Each cp process model is identified by an identification number which is used in the rule. The format of the bitemporal ECA rules is given as follows:

```
ON: modified_Event
IF: (modification_type = 'specifiedModificationType' AND
     modification_scope = 'specifiedModificationScope' AND
     NoD_of_EventContent = 'specifiedNoD_ec' AND
     optional_Additional_Condition)
THEN: cp_Process_ID
```

The ON (trigger) part of the rule is basically the event part of an ECA rule with the exception that the rule is not activated with the occurrence of an event, but with the modification of an event. The condition and action parts of a bitemporal rule have the same semantics as that of an ECA rule. Each part of the rule may have more than one element. These elements can be separated by logical connections such as AND ($\wedge$) and OR ($\vee$). The conditional part of the rule usually has more than one element and these are connected by logical connectors as shown in the rule format above. Logical connectors can also be used when the trigger or action part of a rule has more than one element.

Two important factors to consider about bitemporal rule specification is *rule prioritisation* and *rule composition*. Bitemporal contingency rules may be specified such that one event modification may trigger two or more rules. Since we cannot determine which rule to activate and which ones to ignore, a strategy needs to be put in place to select the appropriate rule. We can use rule prioritisation to indicate the order in which rules should be prioritised. Prioritisation is done using numbers. A higher number signified a lower priority and a lower

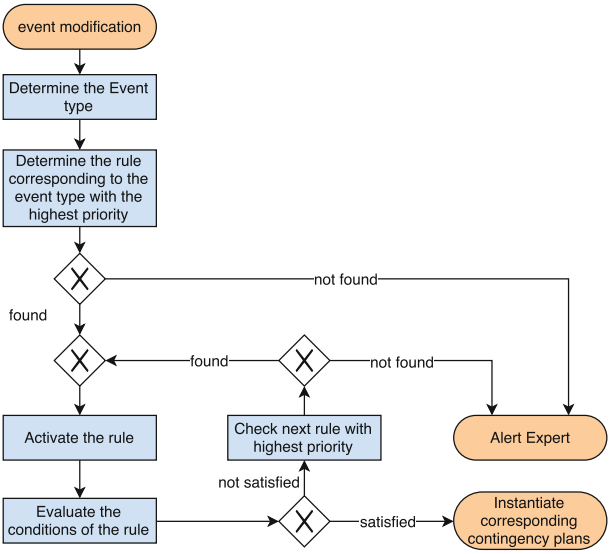



**Fig. 6.** Runtime contingency plan selection

number signifies a higher priority. When numbers are not used, rules are prioritised from top to bottom. When a rule with a higher priority is evaluated and its conditions are not satisfied, the next rule with the highest priority is evaluated.

Rule composition is the combination of two or more rules into one rule provided that each individual rule has the same trigger. Constructs for capturing alternative paths are used in rule composition. We use *else if* construct to facilitate rule composition. The structure of composite bitemporal contingency planning rules is similar to that of nested conditional statements in programming languages. Prioritisation is inherently captured in rule composition. The individual rules are nested in the main rule in order of priority, that is, the condition of the rule with the highest priority is added to the first 'if' condition and the next 'else if' conditions are ordered in a similar manner. Rule composition is an alternative for rule prioritisation with the following benefits: it reduces the number of rules and omits the need to use numbers to prioritise rules. The drawback of rule composition is that it becomes too complex when the composition becomes too large. In such cases, it becomes difficult to read and debug rules. Rule prioritisation can be used alongside rule composition to enhance readability.

### 5.2 Runtime

Ideally, the process model should be sufficient to achieve the organisation's goals. The execution of the process model follows the specification provided during design time passively. When contingencies occur, the process may deviate from its original specification in order to mitigate the effect of the contingency on

the process. CPs are triggered as a deviation from normal process execution to handle contingencies.

When a contingency occurs, a process actively searches for a contingency plan that is suitable for handling the contingency. In Fig. 6, we show how contingency plans are selected during runtime. When a change in the scheduled time of an event is detected we, first of all, determine the event's type. We do this because contingencies are specified for event types. Next, we determine the bitemporal contingency rule with the highest priority associated with that event type. If no rule is found, then a domain expert is alerted about the contingency and it is manually handled. On the other hand, if a rule is found the rule is activated and its conditions evaluated. Once the conditions are satisfied, the corresponding contingency plans are executed automatically. If the conditions for a rule are not met, then the next rule specified for the event type with the highest priority is evaluated. This is repeated until a rule satisfies the conditions. If no such rule exists, then an expert is contacted to manually handle the contingency.

## 6 Discussion

The distinction made between $NoD_{st}$ and $NoD_{ec}$ is important because of the significance of the event's scheduled time. While the scheduled time may be part of an event's content, the distinction is made to properly categorise and handle scheduled time changes. From related work, most changes occur to the scheduled time of an event [7,10,20,26]. Either $NoD_{st}$ or $NoD_{ec}$ may be absent in the conditions of a bitemporal ECA rule. This separation mitigates the difficulty in determining possible contingencies and designing their corresponding CPs.

This work forms part of our ongoing project which consists of (1) detecting event modification in business processes, (2) propagating event modification to other events depending on their causal relationship, (3) managing mutable event provenance and consequently, (4) managing contingencies resulting from event modification in business processes. We have handled some of these aspects in our earlier work [25].

Our implementation of this project so far handles detection, propagation, and provenance of event modification in business processes. This part of the implementation is based on Domain Specific Modelling Environment (DoME)[3]. The DoME tool-set is an extensible collection of integrated model-editing, meta-modelling, and analysis tools supporting a Model-Based Development approach to system and software engineering. DoME uses a *domain specific modelling* approach to problem-solving, i.e., the solution is specified by explicit use of the concepts in the problem domain. DoME is the only open source meta-modelling tool which allows the definition of arbitrary diagram notations using a visual meta-class notation. It also features code generation from visual meta-models, thus, simplifying the implementation of software. The implementation is done

[3] http://dome.ggrossmann.com/.

in Cincom's VisualWorks Smalltalk system. The bitemporal ECA rules responsible for triggering contingency plans during runtime form the back-end to our implementation. The complete prototype is being currently being developed.

## 7 Related Work

Most research approaches in event-driven BPM consider events to be first class citizens and the main driving component of business processes [3,5,11]. In these approaches, events are considered to be immutable once they occur. They lean towards the ideal world scenario and are not equipped to handle situations where revision of an event's content or its scheduled time is required. To the best of our knowledge, there are no approaches in BPM that support the management of bitemporal mutable event.

Substantial research has been done in BPM to enhance process flexibility and adaptability [1,15,17,22]. These approaches can be adopted for modelling contingency plans. The drawback here is that they are not typically designed to handle event correction, but only focused on the other types of triggers, such as an approaching deadline [13], a violated temporal constraint [22], adaptation for different process scenarios [2,16], or compensation in socio-technical processes [8]. We introduce new triggers for process flexibility by incorporating contingency plans that are designed specifically to handle event modification. The practicality of our approach to the designing contingency plans, therefore, relies on existing flexibility approaches in current literature.

Revising an event's content has been considered in some research outside the scope of business process management. In bitemporal database monitoring, event mutability is considered, where an event is modified for the purpose of error correction or new information augmentation [10,20]. Sripada [20] used an event calculus approach to capture event modifications. In their approach, an event recorded in a database has a belief period starting from the transaction time of the event, i.e., when the event is entered into the database. However, the belief period is terminated when a new event revises (corrects) the original event. Belief periods can be used to determine valid events at a particular time. A similar approach to [20] is discussed in [10], where they introduce the concept of multi-history (i.e., capture both the transaction and valid time) of events. In both [10,20], they consider future events (event expected to occur in the future) in their approaches. Their approaches are capable of capturing both proactive and retroactive changes to events. While their work hints towards content modification of events, they replace the entire event with a new event, i.e., the entire content of one event is replaced with the content of a new event. These approaches are outside the scope of BPM. Our work does not focus on belief periods of events as the current state of an event is the state that holds. Instead, we focus on the handling event modification in business process by synthesising contingency plans.

In the domain of web event monitoring, Furche et al. [7] provides an approach for handling web event information update. They focus on improving real

world decision making by monitoring web events and their modifications. Web events (announcements) are prone to constant modifications due to frequent web updates. Decisions in the real world, based on web events, should be capable of maintaining consistency with web events after modification. Similar to other approaches that consider future events, they consider the possibility of both proactive and retroactive changes to web events. This work is outside the scope of BPM but forms the basis for our work. Our work extends these approaches in the web domain to BPM. We focus on modification of business events and how contingency plans can be synthesised to handle such changes. We introduce bitemporal ECA rules suitable for triggering contingency plans during runtime. This makes our work is significantly more suitable for BPM.

## 8 Conclusion

We presented an approach for handling contingencies resulting from event modification in business processes. The contingencies result from either changes in the scheduled time of an event or changes in the event's content. We focus on determining the scope of contingencies, developing contingency plans and bitemporal ECA rules for triggering the contingency plans. The rules are triggered if the scope of a contingency matches the conditions of the rule. When no rule matches the scope of a contingency, a domain expert is alerted. In the future, we hope to integrate the techniques introduced here with our existing work [25] to develop a complete prototype.

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

## References

1. Ayora, C., Torres, V., Reichert, M., Weber, B., Pelechano, V.: Towards run-time flexibility for process families: open issues and research challenges. In: Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 477–488. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36285-9_49
2. Bucchiarone, A., Marconi, A., Pistore, M., Raik, H.: Dynamic adaptation of fragment-based and context-aware business processes. In: Proceedings of ICWS, pp. 33–41, June 2012
3. Buchmann, A., Appel, S., Freudenreich, T., Frischbier, S., Guerrero, P.E.: From calls to events: architecting future BPM systems. In: Barros, A., Gal, A., Kindler, E. (eds.) BPM 2012. LNCS, vol. 7481, pp. 17–32. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32885-5_2
4. da Costa, L.A.G., Pires, P.F., Mattoso, M.: Automatic composition of web services with contingency plans. In: Proceedings of ICWS, pp. 454–461, July 2004
5. Damaggio, E., Hull, R., Vaculín, R.: On the equivalence of incremental and fixpoint semantics for business artifacts with guard-stage-milestone lifecycles. Inf. Syst. **38**(4), 561–584 (2013)

6. Donaldson, L.: The Contingency Theory of Organizations. Sage, Thousand Oaks (2001)
7. Furche, T., Grasso, G., Huemer, M., Schallhart, C., Schrefl, M.: PeaCE-Ful web event extraction and processing as bitemporal mutable events. ACM Trans. Web **10**(3), 16:1–416:7 (2016)
8. Gou, Y., Ghose, A., Chang, C.-F., Dam, H.K., Miller, A.: Semantic monitoring and compensation in socio-technical processes. In: Indulska, M., Purao, S. (eds.) ER 2014. LNCS, vol. 8823, pp. 117–126. Springer, Cham (2014). doi:10.1007/978-3-319-12256-4_12
9. Guide, V.R., Jayaraman, V., Linton, J.D.: Building contingency planning for closed-loop supply chains with product recovery. J. Oper. Manag. **21**(3), 259–279 (2003)
10. Kim, S.-K., Chakravarthy, S.: Temporal databases with two-dimensional time: modeling and implementation of multihistory. Inf. Sci. **80**(1), 43–89 (1994)
11. Montali, M., Maggi, F.M., Chesani, F., Mello, P., van der Aalst, W.M.P.: Monitoring business constraints with the event calculus. ACM TIST **10**, 17:1–17:30 (2013)
12. Niehaves, B., Poeppelbuss, J., Plattfaut, R., Becker, J.: BPM capability development - a matter of contingencies. Bus. Process Manag. J. **20**(1), 90–106 (2014)
13. Pichler, H., Wenger, M., Eder, J.: Composing time-aware web service orchestrations. In: Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. LNCS, vol. 5565, pp. 349–363. Springer, Heidelberg (2009). doi:10.1007/978-3-642-02144-2_29
14. Ploesser, K., Peleg, M., Soffer, P., Rosemann, M., Recker, J.C.: Learning from context to improve business processes. BPTrends **6**(1), 1–7 (2009)
15. Reichert, M., Rinderle-Ma, S., Dadam, P.: Flexibility in process-aware information systems. In: Jensen, K., van der Aalst, W.M.P. (eds.) Transactions on Petri Nets and Other Models of Concurrency II. LNCS, vol. 5460, pp. 115–135. Springer, Heidelberg (2009). doi:10.1007/978-3-642-00899-3_7
16. Rosemann, M., Recker, J.: Context-aware process design: exploring the extrinsic drivers for process flexibility. In: Proceedings of BPMDS@CAISE (2006)
17. Schonenberg, H., Mans, R., Russell, N., Mulyar, N., van der Aalst, W.: Process flexibility: a survey of contemporary approaches. In: Dietz, J.L.G., Albani, A., Barjis, J. (eds.) CIAO!/EOMAS 2008. LNBIP, vol. 10, pp. 16–30. Springer, Heidelberg (2008). doi:10.1007/978-3-540-68644-6_2
18. Snodgrass, R.T.: Temporal databases. In: Frank, A.U., Campari, I., Formentini, U. (eds.) GIS 1992. LNCS, vol. 639, pp. 22–64. Springer, Heidelberg (1992). doi:10.1007/3-540-55966-3_2
19. Sousa, R., Voss, C.A.: Contingency research in operations management practices. J. Oper. Manag. **26**(6), 697–713 (2008)
20. Sripada, S.M.: A logical framework for temporal deductive databases. In: Proceedings of VLDB, pp. 171–182 (1988)
21. Trkman, P.: The critical success factors of business process management. Int. J. Inf. Manag. **30**(2), 125–134 (2010)
22. van der Aalst, W.M.P., Rosemann, M., Dumas, M.: Deadline-based escalation in process-aware information systems. Decis. Support Syst. **43**(2), 492–511 (2007)
23. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features - enhancing flexibility in process-aware information systems. Data Knowl. Eng. **66**(3), 438–466 (2008)

24. Wondoh, J., Grossmann, G., Gasevic, D., Reichert, M., Schrefl, M., Stumptner, M.: Bitemporal support for business process contingency management. In: Jeusfeld, M.A., Karlapalem, K. (eds.) ER 2015. LNCS, vol. 9382, pp. 109–118. Springer, Cham (2015). doi:10.1007/978-3-319-25747-1_11
25. Wondoh, J., Grossmann, G., Stumptner, M.: Propagation of event content modification in business processes. In: Sheng, Q.Z., Stroulia, E., Tata, S., Bhiri, S. (eds.) ICSOC 2016. LNCS, vol. 9936, pp. 70–84. Springer, Cham (2016). doi:10.1007/978-3-319-46295-0_5
26. Wondoh, J., Grossmann, G., Stumptner, M.: Utilising bitemporal information for business process contingency management. In: Proceedings of APCCM, pp. 45:1–45:10. ACM (2016)