Chapter 6
Conformance Verification of Clinical Guidelines in Presence of Computerized and Human-Enhanced Processes

Stefano Bragaglia, Federico Chesani, Paola Mello and Marco Montali

Abstract Clinical Guidelines (CGs) capture medical evidence and describe standardized high quality health processes. Their adoption increases the quality of the service offered by health departments, with direct advantage for treated patients. However, their application in real cases is often tempered by a number of factors like the context, the specific case itself, administrative processes, and the involved personnel. In this chapter we analyse the issues related to the problem of representing CGs in a formal way, and to reason about the differences between what is prescribed by CGs, and what is observed during their application/execution. Our approach is based on a general, abstract framework that should be flexible enough to cope with the raised issues. Possible technical solutions are also presented and their limits discussed.

6.1 Introduction

Clinical Practice Guidelines (or simply Clinical Guidelines, CGs), in their original definition, are "systematically developed statements to assist practitioner and patient decisions about appropriate health care for specific clinical circumstances" [14]. Nowadays, the focus of CGs has been broader to any aspect related to the health care processes, from disease diagnosis, to treatment and intervention, up to administrative issues for health-related services.

Based on medical evidence, CGs provide (1) definitions and terminology, (2) workflows, (3) rules, and (4) temporal constraints. They aim to capture evidence-based new findings and to bring advances into daily medical practice. Their adoption ensures an increase of services quality, and promote the standardization of the health processes across different organizations (at the local, regional, or national level). CGs are also closely related to Clinical Pathways (CPs), that differ from CGs "as they are utilised by a multidisciplinary team and have a focus on the quality and co-ordination of care"[1].

[1] See http://www.openclinical.org/clinicalpathways.html.

Thanks to the pervasive diffusion of IT systems both the CGs, as well as the log of their application to each specific patient, have become available in an electronic form, thus prospecting the possibility of automatically confront the CGs models with what happened in real cases. Hence, the evaluation of how "things have gone" w.r.t. "how things should have gone", named as *conformance*, has become a required analysis step to revise, update and adapt the CGs.

Evaluating the conformance of a CG execution against the CG model however raises a number of technical problems, ranging from the formal representation of CGs and execution logs, up to the reasoning techniques used to establish if and when a CG has not been respected. The aim of this chapter is to analyse all these issues, and to discuss an abstract framework powerful enough to cover many aspects. We do not provide any technical, complete solution: the "final word" on the CGs conformance is far from being achieved. However, we briefly point out how some techniques that can be successfully exploited to overcome many of the current issues.

6.1.1 What Is "Conformance", and Why?

In the context of business processes *conformance* is a property of an observed execution of a *process* (i.e., an *instance* or a *case* of the process), when confronted with a certain *process model*. Conformance indicates if and how much an instance *adheres* to a process model, where such model (explicitly or implicitly) brings some prescriptive information about allowed and forbidden characteristics of process instances.

Thus, a proper definition of conformance depends on three notions: the *instance* of a process, the *model* of a process, and a *matching function* that computes how much an instance matches the model. Definitions of these concepts can vary greatly, depending on the domain and the context. E.g., in a specific hospital department an instance might be the set of actions and events related to a specific patient within a hospitalization; the process model might be a CG that should be applied to that patient; finally, the evaluation function might be a measure of which activities where envisaged by the CG, that were not applied to the patient. The *conformance verification task*, also named as *conformance checking*, amounts to apply the evaluation function to a given instance and to a given model. Usually, it is applied to a number of instances w.r.t. the same model.

Conformance verification is a fundamental and required step whenever a proper analysis of a process is conducted. Indeed, for a variety of reasons, process executions often deviate from the expected model. The conformance verification task answers to the questions: *"Does a case deviate? Where?"*. Notice that from the process management viewpoint some deviations are indeed desirable, while others are to be avoided: the concept of deviation itself does not have a negative meaning, neither a positive one. For example in [13] the authors introduce the distinction of deviations as (acceptable) *exceptions*, versus (undesirable) *anomalies*. Deciding if a deviation is acceptable or not is up to the process manager, that can decide for example to extend the process model to allow/forbid new (previously unforeseen) cases.

6.1.2 Why Executions of Clinical Guidelines Deviate from the CG Model?

CGs capture medical evidence on the basis of statistical data, thus making (at least) three strong implicit assumptions [6]:

(i) *ideal patients*, i.e., patients that have "just the single" disease considered in the CG (thus excluding the concurrent application of more than one CG), and are "statistically relevant" (they model the typical patient affected by the given disease), not presenting rare peculiarities/side-effects;
(ii) *ideal physicians* executing the CG, i.e., physicians whose basic medical knowledge always allows them to properly apply the CGs to specific patients;
(iii) *ideal context* of execution, so that all necessary resources are available.

Moreover, when adopted within local organizations, CGs are typically subject to an adaptation process, which customizes the CG w.r.t. the peculiarities of the specific organization.

Hence, when concretely applying the CGs, three types of issues might arise:

(a) the implicit assumptions $(i) - (iii)$ might not hold, for various practical reasons;
(b) depending on specific contexts, peculiar additional rules and workflows might need to be enacted together with the CGs recommendations (e.g., a certain health department might have its own administrative workflows); consequently, the patient could be subjected to practices not envisaged in the original guideline;
(c) when applying a CG to a specific case at hand, the physician (and the other health-related professionals) exploits also her/his general knowledge (Basic Medical Knowledge, BMK from now on). The interplay between these two types of knowledge can be very complex: e.g., actions recommended by a CG could be prohibited by the BMK, or a CG could force some actions despite the BMK discouraging them.

Note that independently of the type ((a), (b) or (c)) of the arising issue, it is always the physician (and other humans actors) that addresses the problems, and has the responsibility of taking decisions. Thus, healthcare processes can be considered an example of a *socio-technical system* [20], where humans interact with devices, manual and automated activities coexist, and human players ultimately need to cope with an unpredictable, highly dynamic environment that requires continuous adaptation.

Summing up, *CGs propose a model, but when dealing with its effective implementation, many factors might deviate the execution course from the model*. In this light, it becomes extremely important to assess what is effectively going on, and relate actual executions with the “ideal” model. Although each case can be considered in its uniqueness, the focus of the conformance task is upon the totality of the CG implementations versus the CG model.

Conformance checking in CG has another important characteristic: it is not an evaluation of the behaviour of the involved personnel. Indeed, the variety and dynamism of the situations does not allow to automatically (algorithmically) evaluate the personnel’s course of actions and taken decisions. However when applying the CG to a patient, the conformance checking might be of interest for the physicians himself, as a sort of decision support. E.g., each deviation captures an aspect of the current state of affairs that differ from the expected model, and can consequently be analysed by domain experts to formulate a corresponding explanation (e.g., by relying on the conformance framework described in [23]).

6.1.3 Organization of This Chapter

In Sect. 6.2 we briefly introduce Clinical Guidelines, in particular by highlighting the type of information (or, better say, the type of knowledge) that CGs usually contain. In Sect. 6.3 we present an abstract framework, where conformance is expressed in terms of expectations of what should happen, and matching functions between observed (logged) events and expectations. Section 6.4 is devoted to discuss in deep detail how to conjugate two different aspects usually found in CGs, i.e. the interplay between procedural and declarative CG prescriptions. Section 6.5 presents some technical solutions, while finally in Sect. 6.7 we discuss the limits of our current approach, and future works.

6.2 Clinical Guidelines

Clinical Guidelines usually come as documents addressing many different aspects related to the health-care processes. In particular, they address a specific disease or pathology, suggesting the best practices that should be followed/enacted by health practitioners. The principal aim is to provide the patient with the best treatment possible, and guaranteeing quality standards at the same time. The majority of clinical guidelines are provided by national and international public health institutions, although it is frequent to have guideline specifications provided at local levels such as hospitals or departments.

When approaching the conformance task some common CGs features can have a huge impact on the understanding of the conformance issue itself:

Different knowledge types within a CG.
CGs often comprise many type of knowledge. In particular, it is quite usual to encounter:

- *Definitions and terminology:* each CG provides definitions for the terms adopted within the CG, so as to limit misunderstandings. Moreover, each CG usually specifies the disease and the type of patient for which the CG is applicable: in other words, it defines the criteria for applying the CG to a certain patient with a certain disease.
- *Workflows:* a CG can define the set of actions, and their correct execution order, in terms of workflows. Indeed, a plethora of languages and projects has been developed to create domain-independent computer-assisted tools for managing, acquiring, representing and executing CGs [12, 29], paying particular attention to the *procedural and control-flow dimension*.
- *Rules:* particular cases and exceptions are often tackled by CG by means of rules. Sometimes these rules can be applied only when inside a specific execution context of the CG implementation; sometimes the rules must be considered valid for the whole duration of the CG.
- *Linguistic labels:* conditions, (patient) features, and criteria are often measured by means of linguistic labels such as, for example, “low”, “medium” and “high”. While linguistic labels fit perfectly with the involved human actors, their translation into algorithms for the conformance task might be not straightforward.
- *Temporal constraints:* usually a workflow already provides (implicitly or explicitly) a set of temporal constraints, in the sense that a workflow clearly establishes a specific order for the actions execution. Moreover, it is quite common to find explicit constraints related to temporal aspects, such as “a certain B action must be executed within X time unit from action A” (relative-time constraints), or “every day at time Y a certain action must be executed” (absolute-time constraints).

Interplay between CGs and BMK.
CGs must not be intended as mandatory: they are “Not prescriptive: don’t override clinical judgement”[2]. Indeed, it happens that a CG execution trace could seem conformant to the CG and not conformant to the BMK, or vice-versa. Actually, both the CG knowledge and the BMK can be defeated, while it is the physician’s own responsibility to prefer a certain course of actions.

Interplay between workflows and rules.
Both CGs and the BMK contain a mix of procedural (workflows) and declarative knowledge (rules). Procedural knowledge comes into play when there is a set of well-accepted, predefined *sequences of operations* that must be followed by the involved stakeholders. Contrariwise, declarative knowledge typically captures *constraints* and *properties* that must be satisfied during the execution, without explicitly fixing how the stakeholders must behave in order to satisfy them. The majority of the approaches available in the literature have focused either on CGs or BMK in isolation, without

[2]http://www.openclinical.org/clinicalpathways.html.

taking into account how they mutually affect each other. Few works have attempted to consider both these aspects at the same time [6, 10].

Human actors are involved.
Indeed, CGs are mostly implemented by human players. This in turn affects the notion of conformance, that must be adapted to the peculiarities of human players. Let us consider, for example, a rule with a temporal deadline such as "every day the patient temperature must be recorded at 3 p.m.". What happens if the temperature is recorded 5 min later? Should we consider it as conformant with the rule, or not? Although only a physician can answer our question, it seems clear that any conformance approach should allow some flexibility: for example, we might expect that if the temperature is taken with a delay of 5 min, then we could consider it as conformant with a *score* of 0.99, while if the delay is above the hour we could lower the score to 0.40. This example points out to the notion of a *grade of conformance*, against the simpler idea of *conformance yes/no*.

6.2.1 A CG Example

Let us consider a real CG, taken from the on-line repository provided by the "National Institute for Health and Clinical Excellence"[3], a public UK organization sponsored by the UK 's Department of Health. In particular, let us consider the "Quick Reference Guide" of the Clinical Guideline 56.[4] The guideline address the emergency treatment for head injuries, as well as subject admission to specific care units.

After few generalities about the document itself, the first section of the guideline quick reference is devoted to provide *definitions* of the terms. E.g., exact definitions of concepts like "infants", "children", and "adults", are given on the basis of the subject's age. Another example is the explanation of the "Glasgow Coma Scale", referred as GCS: no exact definition is given this time, possibly because GCS is assumed as being part of the staff's Basic Medical Knowledge.

The document then provides a mix of algorithms and rules for dealing with specific tasks; in Fig. 6.1 we show a simple excerpt from the "Assessment in the emergency department" section. There is a procedural part, that is guided by the evaluation of the GCS score: depending on the value assumed at the start of the assessment, different actions should be followed. In this particular case all the paths lead to assess "the need for CT imaging of head and/or cervical spine": such assessment is defined later in the guideline quick reference as another workflow. Moreover, the small CG excerpt in Fig. 6.1 shows at least two rules: the first one is a general rule about stabilizing airway, breathing and circulation (ABC). Another rule instead is a recommendation about "excluding significant brain injury before ascribing depressed conscious level to intoxication", hence suggesting that prior to formulate intoxication, involved personnel should exclude the hypothesis of brain injury.

[3]NICE, http://www.nice.org.uk.

[4]CG56: http://www.nice.org.uk/nicemedia/live/11836/36257/36257.pdf.

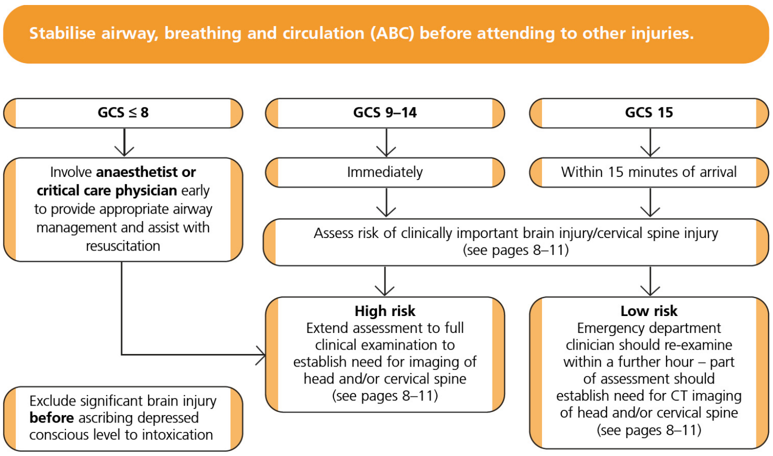

Stabilise airway, breathing and circulation (ABC) before attending to other injuries.
GCS ≤ 8
GCS 9–14
GCS 15
Involve anaesthetist or critical care physician early to provide appropriate airway management and assist with resuscitation
Immediately
Within 15 minutes of arrival
Assess risk of clinically important brain injury/cervical spine injury (see pages 8–11)
High risk
Extend assessment to full clinical examination to establish need for imaging of head and/or cervical spine (see pages 8–11)
Low risk
Emergency department clinician should re-examine within a further hour – part of assessment should establish need for CT imaging of head and/or cervical spine (see pages 8–11)
Exclude significant brain injury before ascribing depressed conscious level to intoxication

Fig. 6.1 An example of a part of a CG taken from UK NICE CG56.

The workflow specifies only few actions. However, interestingly, specific deadlines are provided depending on the different path: in one case, for example, assessment of brain injury must be performed *immediately*, while in another case it can be performed within *15 min*.

Finally, notice that although it is not explicitly specified, the workflow assumes the existence and availability of a "anaesthetist or critical care physician": i.e., the CG envisages certain roles, and assumes that qualified personnel is available for playing the roles.

6.3 A Generic Conformance Framework

In this section, we outline the main components and features of a generic framework for evaluating conformance, following the abstract schema of Fig. 6.2.

6.3.1 Types of Processes and Their Impact on Conformance

In Sect. 6.1.1, we discussed that conformance is about three elements: a *process model*, a *process instance*, and a *matching function*. In particular, there exist many different types of processes. However, from the conformance viewpoint, some process types have a huge impact on the notion of conformance itself.

Open versus Closed processes.
Closed, structured processes are based on the assumption that the model is completely defined, i.e. it explicitly captures all the possible situations. Therefore, any course

of action that is not mentioned in the model is forbidden. Any slightly difference observed within the execution must be intended as a deviation (non conformance) w.r.t. the model. An example of such type of processes is given by bank transactions, where the only and exactly allowed actions are those envisaged by the model.

On the opposite, open processes explicitly define what is allowed, and what is prohibited. Courses of actions for which no information is given are allowed and not required. Open approaches usually adopt a constraint based solution, where a trace is considered conformant if it satisfies all the imposed constraints. Open process are typical of many human interactions, where the involved players can enact many actions not necessarily envisaged by the model.

Summing up, closed approaches require only to specify the desired/allowed actions or events, while open solutions require (at least) two distinct concepts: one for desired actions and one for prohibited actions.

Open- versus Closed-time-view processes.
In principle, the conformance task can be applied "post-mortem", i.e. to already completed executions of the CG, or at run-time, when the execution is still running. This dichotomy needs to be reflected in the adopted conformance checking technique. In the first case, the course of events characterizing the execution is "closed", expected events can be missing either because they did not happen at all, or because they did happen but they were not properly recorded in the underlying information system. In the open-time-view case instead the course of events is "open", since further events can still occur in the future. In principle, this could make some deviations to be only temporary deviations, i.e., apparent deviations that will be fixed thanks to the suitable, future occurrence of new events. The ability of dealing with this open-time-view is typical of runtime verification and monitoring facilities.

With respect to this dimension, a careful consideration must be taken about the real implementations of CGs. In the majority of the cases it happens that health

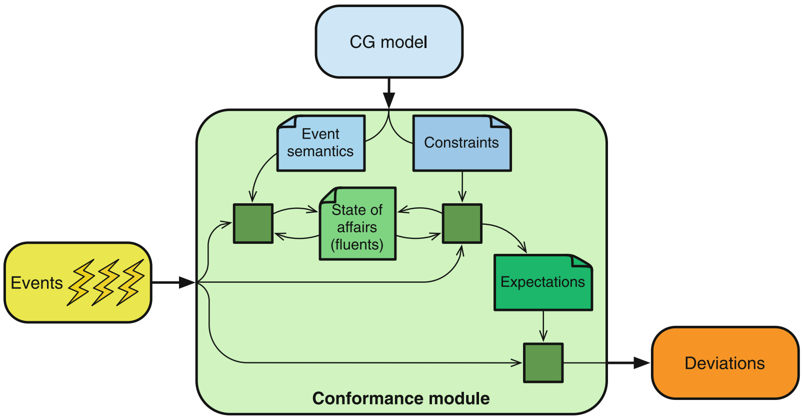


Fig. 6.2 An abstract architecture for conformance

practitioners apply a CG (by executing the foreseen actions), and only at a later moment they record the actions course. This means that two different time instants can be observed: the time instant when an action is executed, and the time instant when the execution of an action is recorded. Such instants are typically referred as *valid time vs. transaction time*. If the difference between these two time instants is relevant, it does not make sense at all to speak about run-time conformance: only a-posteriori, post-mortem analysis can be performed.

6.3.2 An Abstract Architecture for Conformance

Let us first consider a generic matching function as a black-box. It takes as input:

1. a formal model of the clinical guideline, covering the aforementioned different types of knowledge;
2. a set of correlated events describing a single (partial or complete) execution trace of the clinical guideline;

and it computes all the *deviations* between the actual behaviour and the ideal behaviour captured by the guideline model. As for conformance, the CG model can be conceived as constituted by two aspects: one providing the *event semantics*, and the other specifying a set of *constraints* that should be respected by the actual behaviours. Statements about the semantics of an event help in understanding how the occurrences of such an event modify the state of affairs, e.g., by introducing or modifying information about the patient ("measure glucose level has the effect of updating the glucose level of the patient"), or by affecting the state of durative activities ("a complete event marks the termination of an active execution of the corresponding activity").

The term "constraints" is here used as an umbrella term for all those parts of the CG model that specify the intended behaviour foreseen by the evidence-based studies and/or the BMK. Here we find the workflow dimension of the guideline, as well as rules dealing with exceptions or representing a portion of the medical knowledge. The combination of constraints with observed events and state of affairs determines which are the events/actions that are expected to be observed (in the future, as well in the past).

6.3.3 Conformance Based on "Expectations"

Our approach to conformance is inspired by the three elements previously introduced, i.e. the *process case*, the *process model*, and the *matching function*. The first concept we introduce is the *happened event*. Events represent the minimal possible observable information. They are the tiniest bit of information that can be recorded within the system. They are made of a description of *what* happened, together with information about *when* they happened. In our perspective happened events are characterized by having a single time-point duration. Durative actions are given in terms of *start* events and *end* events. Obviously, for each durative action with start time t_s and end time t_e, it must hold $t_s \leq t_e$.

$$\text{Happened Event} \doteq \langle \text{Event Description, When} \rangle$$

Depending on the type of event, the description of the observed event might contain different, structured data fields: in case of the execution of actions, the description might contain the name and the role of the action originator, as well as the name and role of the action destination; in case of observed facts, the description might contain some data that have been observed.

Given the simple notion of event, we introduce in a straightforward manner the notion of a *process instance*, intended as a set of (not necessarily ordered) process events:

$$\text{Process Case} \doteq \{ev_i | i \in 1 \dots n\}$$

where n is the number of events belonging to that instance. Although it is possible to have process cases of infinite length, quite often real cases resort to a finite number of events. The choice of closed- vs. open-time-view processes affects the intended meaning of the process trace: respectively, it contains all the happened events, or rather only a subset of a larger process case.

The second fundamental concept is the *expectation*. The desired behaviour, i.e. the ideal model specified by the CGs, can be expressed by means of expectations, that again are made of *what* is expected, and *when* it is expected. With the term "expectations" we want to capture the notion that depending on the current (dynamic) state of the observed CG execution, the CG model indicates what has to be done and possibly observed. The *what* can be only partly specified, thus allowing to capture larger sets of possible future outcomes. E.g., we might want to expect that a patient is served with food, but we might not want to explicitly specify who is going to serve him, since anyone is fine provided the patient is fed. The *when* instead can be an exact time value, or rather as a time interval when any event happening at a time instant belonging to the interval is fine. E.g., we might expect that patient temperature is taken exactly one hour after the last measurement, or we might expect that the temperature is taken within three hours since the last visit.

$$\text{Expected Event} \doteq \langle \text{What, When} \rangle$$

Depending on the adopted process model (open or close), expectations can be only *positive* (i.e., about the happening of something), or they can be also *negative* (about the non-happening of something, or simply prohibition). Moreover, expectations can be about the happening of events, or rather about properties. In the latter case, expectations can be about *achievement properties*, i.e. about a property being true in a certain time instant; or they can be about *maintenance properties*, i.e. about a property being maintained "true" along a temporal interval. Note that negative expectations implicitly introduce universal quantification over time intervals. Indeed, expecting that a certain event (or property) does not happen within a time interval means that *for all* the time instants within the interval, the event does not happen.

Given the concept of expectations, a process model can be thought as a specification of what is expected for any possible process case. The expressivity of the language used for defining such specifications defines the complexity of the processes that can be modelled.

Finally, the third fundamental concept is the *matching function* that is used to determine if an event or a property matches an expectation. The matching function provides the reasoning capabilities needed to fully support the different knowledge types that are found in a CG, as explained in Sect. 6.2. If we want to support definitions and terminology, we might expect the matching function to support ontological reasoning. If we need to support linguistic labels and grades of conformance, then the matching function should support fuzzy reasoning as well as uncertainty. Minimum capabilities of temporal reasoning are required to establish if an event indeed happened within the expected time interval or not. Finally, a sort of a fuzzy temporal reasoning is required to cope with deviations typically introduced when a process (a CG) is enacted by human players.

Roughly speaking, given *events* and *expectations*, conformance can be established by simply looking which expectations are "satisfied", and which not. To this end, a positive expectation is satisfied if there is an event (a property) that *matches* the expectation, while it is violated if there is not such event. On the contrary, a negative expectation is satisfied if there is no event (property) matching the expected *what/when*, while it is violated if a matching one is found.

Given the concepts of events, expectations and matching functions, only one important question is still open: how a CG (plus a BMK and other rules, standards etc.) can be represented so as to support these concepts? Our current answer is given by tackling in the next Sections three different sub-problems, that we believe as the being the principal issues when dealing with CG: (*a*) the integration between procedural and declarative knowledges; (*b*) representing and reasoning on the state of

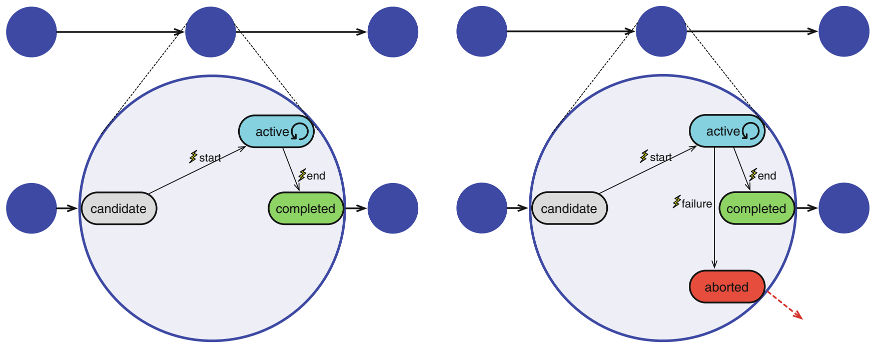


Fig. 6.3 A simple activity life-cycle, and an extension towards exception management

execution of a CG, and how the happening of events affects such states and properties; and (*c*) how to represent declarative knowledge in terms of rules.

6.4 The Interplay Between Procedural and Declarative Knowledge

Clinical Guidelines typically embrace both a procedural, workflow-like dimension, and a more declarative, rule-based component. The first aspect deals with structured, prescriptive and well-established fragments of the guideline, such as for example administrative processes or laboratory procedures. The second instead focuses on the management of less structured fragments of the guideline, as well as with general rules (such as the ones coming from the BMK) that should be always respected during the CG execution. In this section, we discuss the conformance problem by first considering the procedural knowledge, then the declarative knowledge, and finally their combination.

6.4.1 Conformance with Procedural Knowledge

The procedural knowledge defined within a CG takes often the form of a structured workflow, with simple blocks representing the actions to be executed, and control-blocks such as parallel execution, and/or splits, etc. Several workflow-like CG specification languages have been proposed in the literature, such as Asbru [4], GLARE [30], and PROForma [28]. Independently of the specific features of the language, as for conformance all such approaches comprise *intra-* and *inter-activity dynamic constraints*.

6.4.1.1 Activity Lifecycle

Intra-activity constraints aim to capture the so-called *activity life-cycle*, which consists of the acceptable orderings among the constitutive events marking the progression of an instance of the activity, and of the corresponding states. Hence, the activity life-cycle is typically represented by a finite state machine, where nodes represent states of the activity, and edges are labelled with events.

A simple life-cycle is shown in Fig. 6.3 (left), using the GLARE language as a basis (but notice that the concept of life-cycle is orthogonal to the specific language at hand). In this example, the life-cycle just specifies that activities are non-atomic, i.e., each activity execution spans over a time window. More specifically, whenever an activity is *candidate* for execution, a *start* event might be observed, marking the initiation of an *activity instance*. The instance is then put in the *active* state, to explicitly testify that it is currently in execution. To mark the completion of the instance, a corresponding *end* event is used. We observe that *start* and *end* are two atomic events, whereas *candidate*, *active* and *completed* are properties representing the activity instance states. The current state of each activity instance constitutes part of the global state of affairs.

The activity life-cycle requires the description of each event to contain at least two pieces of information: the event type, and the activity it is associated to. Notice also that, in principle, multiple instances of the same activity can be generated, each being associated to a specific instantiation of the corresponding life-cycle. In this work, we make the following assumption:

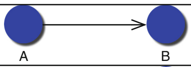
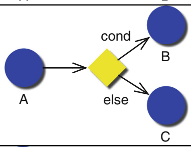
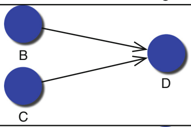
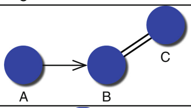
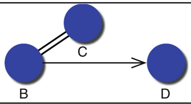
> For each activity, at a given moment in time at most one instance of that activity can be active.

This assumption is motivated by the fact that most activities in the CG refer to the patient, and it is unlikely that the patient is subject to two distinct instances of the same activity at the same time. Obviously, due to the presence of loops and repetitions in typical guidelines, multiple instances of the same activity could occur within a single instance. However, they will be associated to non-overlapping time windows. A further discussion on this assumption is provided below.

Checking conformance with the activity life-cycle when an event is processed breaks down to the following steps:

1. Correlate the happened event and the corresponding life-cycle instance (this comprises the creation of a new instance if a certain event occurs).
2. Check fulfilment of the "next transition" expectation: an event is accepted by the correlated life-cycle instance if it is associated to one of the outgoing transitions from the current instance state.

Table 6.1 Basic workflow patterns in GLARE, and their corresponding enabling conditions

PATTERN	REPRESENTATION	ENABLING CONDITIONS
Sequence		When *A* is completed, *B* becomes candidate
Exclusive choice		When *A* is completed and *cond* holds, *B* becomes candidate When *A* is completed and *cond* does not hold, *C* becomes candidate
Simple merge		When *B* is completed, *D* becomes candidate When *C* is completed, *D* becomes candidate
Parallel split		When *A* is completed, *B* and *C* become candidate
Synchronization		When *B* and *C* are completed, *D* becomes candidate

3. "Advance" the life-cycle instance, moving it from the current state to the next state, following the transition that corresponds to the processed event; this is executed only if the event fulfils the "next transition" expectation.

6.4.1.2 Workflow Constraints and Candidate Activities

The activities of the CG model are usually related to each other by means of inter-activity dynamic constraints, separating the allowed orderings of execution from the forbidden ones. With a procedural flavour, these constraints take the form of a workflow-like structure, which interconnects the activities by means of control-flow patterns [2] such as sequence, choice points, and parallel sections. The richness of such primitives depend on the chosen CG modelling language. A comparative evaluation of some CG modelling languages w.r.t. workflow-patterns support can be found here [22].

In general, as an execution of the CG evolves over time, the workflow determines which are the currently enabled activities, i.e., the *candidate* activities that can/must be executed next. Table 6.1 depicts the five basic control-flow patterns, their representation in GLARE, and their semantics in terms of enabling conditions, i.e., conditions that determine how the corresponding pattern enables some activity when some other activity is completed. As shown in Fig. 6.3 (left), a sequence flow departing from an activity is implicitly connected to its *completed* state, while a sequence edge pointing to an activity is implicitly connected to its *candidate* state. The intuitive semantics sketched in Table 6.1 works thanks to the assumption, stated above, that two instances of the same activity do not overlap. The presence of multiple parallel instances of the same activity would require complex correlation mechanisms to properly apply the control-flow patterns. These mechanisms are typically enforced by the process enactment engine. They rely on internal information that is not relevant for the domain per sé, and that is consequently not guaranteed to be traced and exploitable for conformance checking.

In this setting, conformance checking of control-flow constraints amounts to:

1. Properly handle the computation of candidate activities, applying the control-flow patterns semantics to the current state of affairs (which includes information about the currently completed activities).
2. Impose and verify the negative expectation about non-candidate activities: only candidate activities can be activated by means of a *start* event.
3. Ensure the proper termination of the CG execution when the trace of events is finished; the proper termination is in turn formalized by the following expectations:
   a. every active activity instance is expected to be completed before the termination;
   b. when the execution terminates, no activity can be candidate for execution.

We observe that, considering the conformance characterization provided so far, the CG procedural knowledge gives raise to a "closed" notion of conformance, where every event that is not explicitly expected is considered as forbidden, and no unforeseen activity can be executed. To partially open the workflow specification, more sophisticated forms of activity life-cycle can be introduced. For example, Fig. 6.3 (right) presents an improved version of the basic life-cycle. The new version contains an additional state and transition, to explicitly account for exceptional situations that require the prompt interruption of the running activity instance. This exceptional transition is associated to a *failure* event that leads the instance to an *aborted* terminal state. In this way, it is possible to "cancel" the execution of an activity instance under critical and exceptional circumstances, still conforming to the CG model. Notice that, in principle, the *aborted* terminal state can be associated to a different sequence flow than the one used for the *completed* state. This feature can be exploited to attach a compensation (sub)process meant to manage the exception. When no compensation process is specified, we make the assumption that the sequence flow departing from the *aborted* state is implicitly the same as the one departing from the *completed* state. The rationale behind this "robustness" principle is grounded on a practical observation about how the health operators apply the workflow part of a CG. It can happen that some actions are interrupted (aborted) for many possible reasons, and yet the execution of the CG is brought forward.

6.4.2 Integration with Declarative Knowledge

We further complicate the life-cycle model so as to enable the possibility of modelling declarative rules and constraints related to the CG, and to combine them with the procedural part. The resulting life-cycle has been first proposed in [5, 6], as a result of a close interaction with doctors and healthcare professionals.

Declarative constraints can be exploited to model underspecified portions of the CG, or to complement the CG with general, background medical knowledge (BMK), typically implicitly used by healthcare professionals to adapt the CG on a per-patient basis. For example, the BMK is employed by a physician when an alternative medicine must be found because the patient is allergic to the one mentioned in the guideline specification, or when a critical situation, threatening the life of the patient, suddenly arises. The combination of these two kinds of knowledge is a challenging task, which cannot be solved by simply isolating portions of the CG model that can be captured with a procedural flavour, and those that are better modelled with a declarative approach[5]. On the other hand, such a combination is required in order to better characterize conformance, and in particular to accept justified deviations instead of reporting them to the medical staff.

In order to show how procedural and declarative knowledge can interact in the case of BMK, we summarize in Table 6.2 some of the real-world examples put forward

[5]This is the typical approach followed in BPM, where the process is split into procedural and declarative fragments, or (macro)activities can be expanded by following a declarative or procedural approach (see for example the ad-hoc subprocess construct in BPMN).

in [5, 6]. These examples attest that the activities enabled by the CG procedural model could be prohibited by declarative rules, or conversely that the procedural part could enforce certain behaviours event if they are discouraged by the BMK. More specifically, three interaction modalities arise from the examples:

1. The CG supports the possibility of choosing among two different treatments, and the BMK acts as a business selection rule that helps in determining the route to be taken.
2. The BMK emends the CG, suggesting a suitable way for (temporarily) replacing the workflow prescriptions when they are deemed to be not applicable.
3. The CG defeats the BMK, imposing the prompt execution of an action even if, according to the BMK, it is in general be dangerous for the patient.

This integration gives therefore raise to a hybrid *semi-open* knowledge, where the procedural CG model must partially support the execution of unpredicted activities, as well as some deviations from its prescriptions, while the BMK must acknowledge the possibility of being defeated, honoring the motto: “domain experts always get the last word”.

A deep understanding concerning the nature of this hybrid knowledge, and how its building components actually interact, is still far to be reached. Nevertheless, a first necessary step towards a proper characterization of conformance in this setting requires to revise the activity lifecycle, so as to reflect this interplay.

The revised lifecycle is shown in Fig. 6.4 (left). It is enriched with additional states and transitions, which are not only associated to events, but also to conditions that

Table 6.2 Examples of clinical behaviors induced by the interplay between procedural recommendations coming from a CG and declarative rules expressing part of the BMK, taken from [5, 6]

	CG	BMK	CG+BMK
A	Patients suffering from bacterial pneumonia must be treated with penicillin or macrolid	Do not administer drugs to which a patient is allergic	Administer macrolid to a patient with bacterial pneumonia if she is allergic to penicillin
B	Patients with post-hemorrhagic shock require blood transfusion	Do not apply therapies that are not accepted by patients. Plasma expander is a valid alternative to blood transfusion, provided that …*(omitted)*	If the patient refuses blood transfusion, in case of post-hemorrhagic shock treat her with plasma expander
C	In patients affected by unstable angina, coronary angiography is mandatory	A patient affected by advanced predialytic renal failure should not be subject to coronary angiography, because the contrast media may cause a further deterioration of the renal functions	Even in case of a predialytic renal failure, perform coronary angiography if the patient is affected by unstable angina

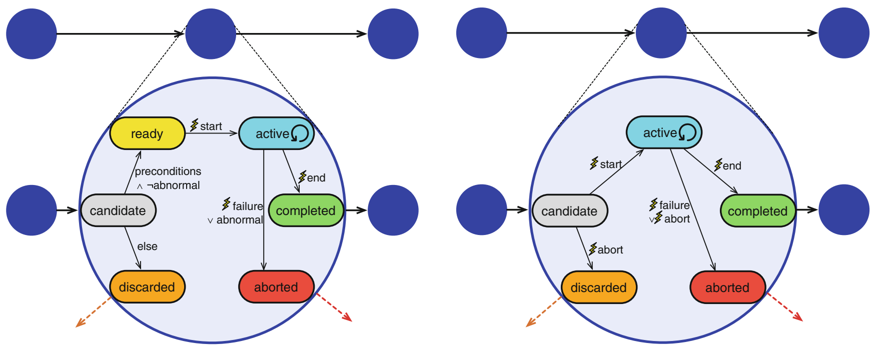


Fig. 6.4 A sophisticated activity lifecycle supporting abnormal situations and exceptions; the left diagram shows the intended lifecycle, and the right one its pure event-based version, which can be reconstructed from the analyzed trace

are checked against the current state of affairs (e.g., to verify whether the patients' data are within certain ranges). When an activity instance is *candidate*, it is not still ready to be executed. To make it *ready* for the execution, some additional conditions must be satisfied. More specifically, to be executed a candidate action must satisfy its *preconditions*, which are a part of the description of the activity. Preconditions specify whether the activity is applicable in the current state of affairs, and are evaluated on the basis of the currently available patients data and execution context. Even though preconditions are satisfied, the action cannot be executed if the current situation is "abnormal". This is captured by means of *abnormality* conditions, which are satisfied whenever the assumptions made in the CG model (e.g., *ideal* patient and context), do not hold. If the situation is not abnormal and preconditions hold, the action is *ready* to be executed. Otherwise, it is *discarded*. A *ready* activity instance can be made *active* by triggering the *start* event. Two cases are possible then: either an *end* event occurs marking that the activity instance is *completed* or an *abnormality/failure* shows up during execution, so that the action is *aborted*. As described before *failure* events mark exceptional situations that require the immediate interruption of the activity instance. The additional abnormality test is instead used to capture those situations in which the activity is started in a "normal" situation, which however becomes abnormal during the execution of the activity.

We observe that the *preconditions* are specified in the (augmented) procedural CG model, the *failure* situations depend on a specific execution, while *abnormality* circumstances are typically identified and handled by means of BMK rules. In addition, further constraints can be imposed by the BMK depending on the current context and patient's status; from the conformance point of view, this means that activity executions unforeseen by the procedural knowledge should be accepted if they are made *candidate* by the BMK.

As for conformance with this revised lifecycle, it is worth noting that its aim is not to enforce a discard/abortion of the activity instance when the corresponding state-related conditions prescribe to do so, but to check whether the actual behaviour is aligned with the ideal one. During the effective execution, healthcare professionals will decide whether the activity instance must be aborted or not, and the conformance checker will evaluate whether this course of execution deviates from the intended transitions or not. To track the actual transitions inside the activity instance lifecycle, we should then ideally replace the pre- and abnormality conditions with a corresponding *abort* event, triggered by the healthcare professionals when they consider the activity to be discarded before its execution, or aborted during its execution. This purely event-based variant is shown in Fig. 6.4 (right). The conformance problem ultimately amounts to detect and report all those instances for which the expected lifecycle transition is deviates from the actual performed transition.

Finally, notice that the *discarded* terminal state induces a third "exit point" from the activity. As for the *aborted* state, this exit point can be explicitly handled by means of additional CG/BMK rules, or be connected by default to the same outgoing sequence flow used for the *completed* state (thus ensuring robustness).

6.5 Representation of Clinical Guidelines

In Sect. 6.3 we introduced the concept of expectation: a CG could be thought of a set of rules and constraints that points out what is expected to happen next. However from the discussion in Sect. 6.4 it appears that expectations alone are not sufficient. Indeed, there is the need to reason upon the "state of affairs" when executing a CG. Such state of affairs is independent of what is expected next, and on the contrary the generation of expectations starts always from the current state. Hence, any CG in our model can be thought of as two distinct yet related descriptions: rules that specify how the happening of events affects the current state of affairs, and rules that forecast what is expected then on the basis of the current state and happened events.

Our current approach exploits two existing solutions: representation and reasoning upon the "state of affairs" is done by means of the Event Calculus [19], while expectations are represented, generated and verified by means of the Event-Condition-Expectation (ECE-) rules [7]. These two approaches can be easily integrated together, since the ECE-rules natively exploit the notion of *fluent*, core concept of the Event Calculus. From the technical viewpoint, the integration can be achieved since there exists implementations of ECE-rules and EC based on the Drools Framework.

6.5.1 Representing the Guideline Evolution with Event Calculus

In 1986, Kowalski and Sergot proposed the Event Calculus (EC, [19]) as a general framework to reason about time, events and change, overcoming the inadequacy of time representation in classical logic. It adopts an explicit representation of time, accommodating both qualitative and quantitative time constraints. Furthermore, it is based on (a fragment of) first-order logic, thus providing great expressiveness (such as variables and unification). Shanahan [25] intuitively characterizes the EC as "a logical mechanism that infers *what is true when*, given *what happens when* and *what actions do*".

The three fundamental concepts are that of *event*, happening at a point in *time* and representing the execution of some action, and of properties whose validity varies as time flows and events occur; such properties are called *fluents*. An EC specification is composed of two theories, each containing a set of axioms:

- a general (domain-independent) theory axiomatizing the *meaning* of the predicates supported by the calculus, i.e., the so called *EC ontology* shown in Table 6.3;
- a domain theory that *exploits* the predicates of the EC ontology to formalize the specific system under study in terms of events and their effects upon *fluents*. Our domain theory is focused on the formalization of intra- and inter-activity constraints, together with the corresponding expectations and deviations.

The domain knowledge about actions and their effects corresponds to "What actions do". The capability of an event to make a fluent true (false respectively) at some time is formalized by stating that the event *initiates* (*terminates*) the fluent. More specifically, when an event e occurs at time t, so that $initiates(e, f, t)$ and f does not already hold at time t, then e causes f to hold. In this case, we say that f is *declipped* at time t. There is also the possibility to express that some fluent holds in the initial state, using the $initially$ predicate. Conversely, if $terminates(e, f, t)$ and f holds at time t, then e causes f to not hold any more, i.e., f is *clipped* at time t.

"What happens when" is the execution trace characterizing a (possibly partial) instance of the system under study. An execution trace is composed of a set of occurred events. The basic forms of EC assume that events are atomic, i.e., bound to a single time point. In particular, an execution trace is composed of a set of *happens* binary predicates, listing the occurrences of events and their corresponding timestamps.

The combination of the domain knowledge and a concrete execution trace leads to infer "what is true when", i.e., the intervals during which fluents *hold*. The $holds\_at(f, t)$ predicate of the EC ontology is specifically used to test whether f holds at time t.

Table 6.3 The basic Event Calculus ontology

Predicate	Meaning
$initially(F)$	Fluent F holds in the initial state of affairs
$initiates(Ev, F, T)$	Event Ev initiates fluent F at time T
$terminates(Ev, F, T)$	Event Ev terminates fluent F at time T
$happens(Ev, T)$	Event Ev occurs at time T
$holds\_at(F, T)$	Fluent F holds at time T
$holds\_from(F, T, T_s)$	Fluent F holds at time T since time T_s

In [6] we introduced for the first time a model of the execution of a single action, hence opening up the possibility to integrate it with the declarative knowledge. We will not report here the technicalities presented in [6]; however, we might point out that few basic fluents such as *status* (indicating possible status of an activity, such as “candidate”, “active”, “completed”, etc.), together with events of “start”, “end”, “discard” and “abort” have been sufficient to fully represent the activity lifecycle discussed in Sect. 6.4.1.

6.5.2 Generating and Matching Expectations with ECE Rules

In several previous works we have explored the notion of expectations, and we have defined several different languages for defining rules that support the definition of expectation. In particular, in [7] we have introduced the Event-Condition-Expectations (ECE-) rules. Based on the rule framework Drools[6], ECE-Rules allow to link current system status (properties, and also Event Calculus fluents) and the dynamic happening of events to the generation of expectations.

An example of a rule is shown in Fig. 6.5. When a patient $\$pat$ is evaluated to be at risk of a disease $\$disease$, with a factor judged as to be “high” and with a confidence equal or greater to the “medium” grade, then it is expected that a proper treatment is initiated within one hour from the evaluation.

This simple rule already shows the power of the ECE-Rules: the rule triggers when the evaluation of the disease risk is inserted as a event and conditions are met. As a consequence, dynamically an expectation is generated. The expectation then can be satisfied by an event representing the start of proper treatment. Note that the formalism allows to define also proper actions in case of satisfaction of the expectations (rewards) and in case of violations (possibly expected countermeasures).

```
rule "Risk factor evaluation"
when
    $pat : Patient( ... )  // patient identifier
    // evaluation of risk factor and confidence degree
    $risk : EvaluatedRisk( $phys, $pat, $disease, $factor, $conf )
    $factor == "high"
    $conf >= "medium"
then
    expect InitiateTreatment( $pat, $disease, this after[0,1hour] $risk )
        on fulfillment {        // if the treatment is initiated
            /* some increase in patient health */
        }
        on violation {         // if the treatment is not initiated
            alert( ... );
        }
end
```

Fig. 6.5 An example of ECE-Rule [7].

[6]http://www.jboss.org/drools/.

The choice of Drools as supporting framework for the ECE-rules is based on the Drools Chance extension, and in particular on the possibility of support various type of imperfect reasoning [26]. Indeed, the possibility of support at least fuzzy logic-based reasoning is fundamental to support the human-related nature of CGs: for example, a number of linguistic qualifiers like "high", "low" or "medium" are usually involved in guidelines specification. Moreover, imperfect reasoning is needed to cope with deadlines, specifically if humans are involved.

6.6 Related Work

There is a flourishing literature focused on conformance issues in the healthcare setting, mainly due to the impact that the execution of CGs has in terms of quality, cost savings, and effectiveness. We review some of the relevant approaches, starting with two observations. First, often the term conformance is replaced by *compliance*, so as to emphasize normative and legal aspects, or by *critiquing*, stressing the fact that the actual courses of execution are critically analysed. Second, an impressive series of works aims at providing specific, vertical solutions tailored to a single guideline or disease, and by no means we can cover this extensive literature here.

In [18], an empirical, interview-based assessment is carried out so as to understand how healthcare professionals perceive the adoption of CGs, and their usage to monitor conformance. Interestingly, the assessment of conformance with the recommendations included in the CGs is perceived by clinicians as an importat problem, with which however they do not have enough familiarity. As a recommendation for future research, they also mention the problem of putting the patient into the loop, understanding to what extent patient concordance with the CG recommendations has to be considered when assessing conformance.

The vast majority of approaches focused on conformance in the clinical setting only considers the contribution of the CGs, without dealing with how they interact with the BMK. In this respect, checking conformance is tightly related to operational decision support and conformance verification in the field of process mining [1], which is being increasingly applied to the healthcare setting [21]. Notable examples of such a cross-fertilization are [15, 16]. In [16], a graphical language for specifying declarative processes is used to capture clinical recommendations, expressing constraints about the relative occurrence of activities, as well as the data they carry. At the same time, a procedural model of a CG is simulated so as to extract possible execution scenarios. The simulated traces are then checked against the formalized recommendations so as to ascertain whether they agree or not. The approach is then extended in [15], where ontologies are exploited so as to reuse the same formalization of clinical recommendations for checking conformance of different CG models belonging to the same Open Clinical repository.

Notice that the notion of conformance used in [15, 16] is radically different from ours, because it uses as input data those extracted through simulation from the ideal

CG model, not real executions. In fact, [15, 16] can be considered as examples of techniques that show how conformance checking as intended in this paper can be complemented with techniques and tools that ascertain conformance/compliance via model checking[7]. On the one hand, formal verification applied a-priori on the CG models, before their actual executions, can help in preventing the presence of errors at runtime, and in improving the quality of models. On the other hand, they are not exhaustive, due to the fact that they do not consider specific unforeseen situations that may arise at runtime, they do not work on real patient data and contexts, and they do not consider how the BMK could be employed to dynamicall adapt the CG models on a per-patient basis. We therefore believe that both approaches are needed, so as to provide support to the healthcare professionals during the entire CG lifecycle. For more details about formal verification and model checking of CGs, the interested reader can refer to the chapter on verification in this book.

Notably, model checking techniques can be suitably employed not only for the a-priori verification of CGs, but also to tackle conformance in the way intended in this chapter. This is the case of [17], where model checking techniques are used to compare ideal actions prescribed by a CG with actual actions extracted from healthcare records that log real executions of the CG. The focus is mainly on the control-flow/temporal dimension, without taking into account resources and event data. Of particular interest is the elicitation of two lists of reasons for non-compliance, singled out by respectively considering the adherence of the actual with the expected behavior, and whether the actual behavior is supported by the patient findings. Since this second class of reasons implicitly depends on the BMK (which is used to indicate and explain why a certain action is (un)likely to be executed given certain patient findings), it would be interesting to encode the different reasons for non-compliance in our approach. This would allow us to not only report deviations back to the healthcare professionals, but also in automatically provide hints about the reasons for such deviations.

Another approach that aims at going beyond the detection of deviations is that of [3]. The authors employ Asbru to model che CG, and describe a technique to check the adherence of an observed execution to the intended model. However, they also consider preferences and policies of the institution in which the guideline is executed, and whenever a deviation is detected, they check whether the deviation can be explained by applying such additional knowledge. In this respect, policies and preferences of the institution can be considered as part of the BMK, paving the way towards the extension of our framework with preference-based reasoning.

The notion of clinical guideline conformance based on a formally defined matching between the actual and the expected behavior started in [8, 11], where the SCIFF framework, based on abductive logic programming with hypothesis confirmation, is applied to clinical guideline conformance, with application to cancer screening protocols. The possibility of exploiting the framework starting from typical procedural CG

[7]Notice that, even though the techniques in [15, 16] are presented as "a-posteriori" techniques, they could be considered as "a-priori" technique, because they work on the CG model, not on its real enactments.

models is tackled in [9], which presents a translation mechanism that analyzes the CG model and produces corresponding SCIFF rules. Differently from the approach here presented, the SCIFF framework tailors non-conformance to logical inconsistency, and it is therefore only able to determine whether a (partial or complete) execution trace complies with the intended model or not, without continuing with the analysis when a deviation is detected.

The general conformance framework here presented generalizes that of [5, 6] along two directions: on the one hand, we provide here a thorough analysis of the main features a generic conformance framework must provide, and on the other hand we describe a more comprehensive activity lifecycle. Interestingly, an alternative approach to our Event Calculus-based one is presented in [27], where Answer Set Programming is used to encode a preliminary version of the activity lifecycle is presented, enumerating with specific rules all the possible types of deviations that may be encountered.

Finally, we would like to point the interested reader to [24], which provides a broad analysis of the role of compliance in the healthcare setting, and its impact on the development of computerized decision-support systems for CGs.

6.7 Discussion

The adoption of Clinical Guidelines is continuously increasing, towards high quality standards in health processes. At the same time, though, healthcare professionals might run into several issues when operating in agreement with CGs, due to unforeseen situations, contextual factors, specific peculiarities of patients, administrative problems, and human decisions. When executing CGs, deviations from the expected behavior are often observed. This by no means imply a negative impact on the patient, but simply attests a discrepancy between the expected and actual execution. Detecting the presence of such deviations is nevertheless of key importance towards CG improvement on the one hand, and awareness of the patient state on the other hand.

Understanding if a CG execution deviates means to evaluate its *conformance* w.r.t to the CG model. However, the nature of CGs makes it a difficult task. One reason resides on the type of knowledge encoded in CGs: definitions, structured workflows, rules, linguistic qualifiers and temporal constraints are usually part of any CG specification. Moreover, such knowledge is expressed using both a procedural approach (e.g., the workflows), and in a declarative way (as it happens with the many rules). A second reason lies on the fact that CG are not prescriptive models: during their execution physicians and personnel continuously integrate CG with Basic Medical Knowledge, hence adding/changing/avoiding actions. A third reason is related to the socio-technical nature of CGs: e.g., deadlines for human beings might have a different semantics from deadlines in fully automated processes.

In this chapter we outlined an abstract framework for dealing with conformance, based on the notion of expectations and of matching function. The more sophisticated the matching function, more complex the type of conformance that can be verified.

However, the double-nature of CGs as being partly procedural and partly based on rules requires a deeper analysis of how these two components can inter-relate. Our proposed approach is based on an extended version of the activity life-cycle, where exceptions can occur and interrupt the execution of a single activity. We have not discussed technical solutions, but we pointed out that existing solutions might cope with the highlighted complexity.

In this chapter we have completely ignored few important dimensions, that indeed in the Business Process field are subject of an intense research activity. First of all, knowing that a deviation happened might not be sufficient: a further question is "why the deviation happened". Strictly related to this point there is the identification of the culprit for the deviation. Answer such question would require to specify in the CG specification also concepts like *responsibility*, *duties*, *permissions*, and other deontic concepts. Another question is about evaluating numerically *how much* the overall process executions deviated from the CG model. A measure of deviation would help to identify the most problematic processes, and to establish if and when corrective measures are needed. Given the specific health domain, it is reasonable to expect that any measurement function must take into account the domain semantics of the actions and of the deviations. Notice that since deviations might have also a positive impact, any measure of deviation should take into account also the produced effects.

Our current work is focused on building a unified framework where the conformance task can be accomplished. However, the nature of the hybrid reasoning techniques required by conformance is proving to be a challenging task: in particular the need of models and algorithms for perfect and imperfect reasoning at the same time is an open problem.

Acknowledgments The approaches presented in this paper are the result of discussions and collaborations with many colleagues. In particular, we would like to thank Davide Sottara, Emory Fry, Paolo Terenziani, Alessio Bottrighi, and Stefania Montani.

This work has been partially supported by the Health Sciences and Technologies - Interdepartmental Center for Industrial Research (HST-ICIR) - University of Bologna, by the DEIS Depict Project, and by the EU FP7 IP project Optique (*Scalable End-user Access to Big Data*), grant agreement n. FP7-318338.

References

1. van der Aalst, W.M.P.: Process Mining - Discovery, Conformance and Enhancement of Business Processes. Springer, Berlin (2011)
2. van der Aalst, W.M.P., et al.: Workflow patterns. Distrib. Parallel Databases **14**(1), 5–51 (2003)
3. Advani, A.A., Lo, K.-K., Shahar, Y.: Intention-based critiquing of guideline-oriented medical care. In: American Medical Informatics Association Annual Symposium (AMIA). AMIA (1998)
4. Balser, M., Duelli, C., Reif, W.: Formal semantics of Asbru-an overview.In: Proceedings of IDPT 2002 (2002)

5. Bottrighi, A., Chesani, F., Mello, P., Molino, G., Montali, M., Montani, S., Storari, S., Terenziani, P., Torchio, M.: A hybrid approach to clinical guideline and to basic medical knowledge conformance. In: Combi, C., Shahar, Y., Abu-Hanna, A. (eds.) AIME 2009. LNCS, vol. 5651, pp. 91–95. Springer, Heidelberg (2009)
6. Bottrighi, A., Chesani, F., Mello, P., Montali, M., Montani, S., Terenziani, P.: Conformance checking of executed clinical guidelines in presence of basic medical knowledge. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011, Part II. LNBIP, vol. 100, pp. 200–211. Springer, Heidelberg (2012)
7. Bragaglia, S., Chesani, F., Fry, E., Mello, P., Montali, M., Sottara, D.: Event condition expectation (ECE-) rules for monitoring observable systems. In: Olken, F., Palmirani, M., Sottara, D. (eds.) RuleML - America 2011. LNCS, vol. 7018, pp. 267–281. Springer, Heidelberg (2011)
8. Chesani, F., et al.: Compliance checking of cancer-screening careflows: an approach based on computational logic. In: Computer-based Medical Guidelines and Protocols: A Primer and Current Trends. Studies in Health Technology and Informatics, vol. 139, pp. 183–192. IOS Press, Amsterdam (2008)
9. Chesani, F., Mello, P., Montali, M., Storari, S.: Testing careflow process execution conformance by translating a graphical language to computational logic. In: Bellazzi, R., Abu-Hanna, A., Hunter, J. (eds.) AIME 2007. LNCS (LNAI), vol. 4594, pp. 479–488. Springer, Heidelberg (2007)
10. Christov, S., et al.: Formally defining medical processes. Methods Inf. Med. **47**(5), 392 (2008)
11. Ciampolini, A., et al.: Using social integrity constraints for on-the-fly compliance verification of medical protocols. In: Proceedings of the 18th IEEE Symposium on Computer Based Medical Systems (CBMS 2005), pp. 503–505. IEEE Computer Society Press (2005)
12. Fridsma, D.B. (Guest ed.): Special issue on workflow management and clinical guidelines. JAMIA **22**(1), 1–80 (2001)
13. Depaire, B., Swinnen, J., Jans, M., Vanhoof, K.: A process deviation analysis framework. In: La Rosa, M., Soffer, P. (eds.) BPM Workshops 2012. LNBIP, vol. 132, pp. 701–706. Springer, Heidelberg (2013)
14. Field, M.J., Lohr, K.N. (eds.): Committee to advise the public health service on clinical practice guidelines. Clinical Practice Guidelines: Directions for a New Program. The National Academies Press, 1990
15. Grando, M.A., van der Aalst, W.M.P., Mans, R.S.: Reusing a declarative specification to check the conformance of different CIGs. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011, Part II. LNBIP, vol. 100, pp. 188–199. Springer, Heidelberg (2012)
16. Grando, M.A., Schonenberg, M.H., van der Aalst, W.M.P.: Semantic process mining for the verification of medical recommendations. In: 4th International Conference on Health Informatics (HEALTHINF), pp. 5–16 (2011)
17. Groot, P., et al.: Using model checking for critiquing based on clinical guidelines. Artif. Intell. Med. **46**(1), 19–36 (2009)
18. Hutchinson, A., et al.: Towards efficient guidelines: how to monitor guideline use in primary care. Health Technol. Assess. **7**(18) (2003)
19. Kowalski, R.A., Sergot, M.J.: A logic-based calculus of events. New Gener. Comput. **4**(1), 67–98 (1986)
20. Kroes, P., et al.: Treating socio-technical systems as engineering systems: some conceptual problems. Syst. Res. Behav. Sci. **23**(6), 803–814 (2006)
21. Mans, R.S., van der Aalst, W.M.P., Vanwersch, R.J.B., Moleman, A.J.: Process mining in healthcare: data challenges when answering frequently posed questions. In: Lenz, R., Miksch, S., Peleg, M., Reichert, M., Riaño, D., ten Teije, A. (eds.) ProHealth 2012 and KR4HC 2012. LNCS, vol. 7738, pp. 140–153. Springer, Heidelberg (2013)
22. Mulyar, N., van der Aalst, W.M.P., Peleg, M.: A pattern-based analysis of clinical computer-interpretable guideline modelling languages. J. Am. Med. Inform. Assoc. **14**, 781–787 (2007)
23. Quaglini, S.: Compliance with clinical practice guidelines. Stud. Health Technol. Inform. **139**, 160–179 (2008)

24. Quaglini, S.: Compliance with clinical practice guidelines. In: Computer-based Medical Guidelines and Protocols: A Primer and Current Trends. Studies in Health Technology and Informatics, vol. 139, pp. 160–179. IOS Press, Amsterdam (2008)
25. Shanahan, M.: The event calculus explained. In: Veloso, M.M., Wooldridge, M.J. (eds.) Artificial Intelligence Today. LNCS (LNAI), vol. 1600, pp. 409–430. Springer, Heidelberg (1999)
26. Sottara, D., Mello, P., Proctor, M.: A configurable rete-oo engine for reasoning with different types of imperfect information. IEEE Trans. Knowl. Data Eng. **22**(11), 1535–1548 (2010)
27. Spiotta, M., Bottrighi, A., Giordano, L., Dupré, D.T.: Conformance analysis of the execution of clinical guidelines with basic medical knowledge and clinical terminology. In: Miksch, S., Riano, D., Teije, A. (eds.) KR4HC 2014. LNCS, vol. 8903, pp. 62–77. Springer, Heidelberg (2014)
28. Sutton, D.R., Fox, J.: The syntax and semantics of the proforma guideline modelling language. J. Am. Med. Inform. Assoc. **10**(5), 433–443 (2003)
29. Ten Teije, A., Miksch, S., Lucas, P. (eds.): Computer-based Medical Guidelines and Protocols: A Primer and Current Trends. Studies in Health Technology and Informatics, vol. 139. IOS Press, Amsterdam (2008)
30. Terenziani, P., et al.: Applying artificial intelligence to clinical guidelines: the GLARE approach. In: Ten Teije, A., Miksch, S., Lucas, P. (eds.) Studies in Health Technology and Informatics, vol. 139, pp. 273–282. IOS Press, Amsterdam (2008)