Conformance Checking of Executed Clinical Guidelines in Presence of Basic Medical Knowledge

Alessio Bottrighi[1], Federico Chesani[2], Paola Mello[2], Marco Montali[3], Stefania Montani[1], and Paolo Terenziani[1]

[1] DI, Univ. del Piemonte Orientale, via Bellini 25/g, 15100 - Alessandria, Italy
{alessio.bottrighi,terenz,stefania}@mfn.unipm.it
[2] DEIS - Univ. di Bologna, viale Risorgimento 2, 40136 - Bologna, Italy
{federico.chesani,paola.mello}@unibo.it
[3] KRDB Research Centre, Free University of Bozen-Bolzano, Italy
montali@inf.unibz.it

Abstract. Clinical Guidelines (CGs) capture medical evidence, but are not meant to deal with single patients' peculiarities and specific context limitations and/or constraints. In practice, the physician has to exploit basic medical knowledge (BMK) in order to adapt the general CG to the specific case at hand. The interplay between CG knowledge and BMK can be very complex. In this paper, we explore such interaction from the viewpoint of the *conformance* problem, intended as the adherence of an observed CG execution trace to both types of knowledge. We propose an approach based on the GLARE language to represent CGs, and on an homogeneous formalization of both CGs and BMK using Event Calculus (EC) and its Prolog-based implementation $\mathcal{REC}$, focusing on "a posteriori" conformance evaluation.

Keywords: Clinical Guidelines, Conformance, Event Calculus, Integration with Basic Medical Kwnoledge.

1 Introduction

Clinical Guidelines (CGs) are, in the definition of the MeSH dictionary, "work consisting of a set of directions or principles to assist the health care practitioners with patient care decisions about appropriate diagnostic, therapeutic, or other clinical procedures for specific clinical circumstances". One of the main goals of CGs is to capture medical evidence and to put it into practice. However, from one side, evidence is essentially a form of statistical knowledge, and is used to capture the generalities of classes of patients, rather than the peculiarities of a specific patient. From the other side, demanding to expert committees the elicitation of all possible executions of a CG on any possible specific patient in any possible clinical condition is an infeasible task. Thus, several conditions are usually implicitly assumed by experts building a CG:

F. Daniel et al. (Eds.): BPM 2011 Workshops, Part II, LNBIP 100, pp. 200–211, 2012.

(*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 allow them to properly apply the CGs to specific patients;
(*iii*) *ideal context* of execution, so that all necessary resources are available.

On the other hand, when a specific physician applies a given CG to a specific patient, the patient and/or the context may not be "ideal". For instance, some laboratory instrument (recommended by the CG) may be missing, and/or the patient may show specific conditions not foreseen in the general CG. As a consequence, the physician has to exploit her/his general knowledge (Basic Medical Knowledge, BMK from now on) in order to adapt the general CG to the specific case at hand. 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.

The issue of studying the interplay between the knowledge in CGs and BMK is a fundamental one, to promote the practical applicability of CGs themselves. However, it is relatively new in the literature, and has not yet been deeply investigated. In the last two decades most approaches have focused either on CGs or BMK in isolation, without taking into account how they mutually affect each other. In particular, a plethora of languages and projects has been developed to create domain-independent computer-assisted tools for managing, acquiring, representing and executing CGs [8,13], paying particular attention to the *procedural and control-flow dimension*.

This observation points out another challenging and relatively unexplored issue: while current approaches capture CGs with a workflow-like modeling style, both CGs and the BMK contain a mix of procedural and declarative knowledge. 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.

In this paper, we explore how CGs workflow-based approaches can be extended to take into account also the BMK, providing a uniform underlying logic-based formalization that is able to accommodate both procedural and declarative knowledge. We explore the interaction between CGs and BMK from the viewpoint of the *conformance* problem, intended as the adherence of an observed CG execution trace to both types of knowledge.

From a formal viewpoint, there are many different definitions of conformance. Procedural approaches typically consider a CG execution trace conformant if it contains all and only the actions envisaged by the specification, in the right order. More flexible declarative solutions usually adopt a constraint based approach, where a trace is considered conformant if it satisfies all the imposed constraints. A central issue is 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, and it is the physician's responsibility to assess if a trace can be deemed as conformant or not. Hence, our aim is to support the physician in the conformance evaluation task, providing her the most information possible, and consequently easing the evaluation process.

The automatic tool we propose in this paper is based on GLARE [14] to represent CGs, and relies on an homogeneous formalization of both CGs and BMK using Event Calculus (EC) and its Prolog-based implementation $\mathcal{REC}$. In particular, we use the EC to represent procedural aspects of CG, while we exploit Prolog clauses to represent the BMK in terms of logic rules. Note that, even if in the paper we focus on CGs, GLARE is able to manage protocols and since our approch is general, it can be applied to protocols in the same way.

The paper is organized as follows: in Section 2 we motivate our work showing the interaction between CG and BMK; in Section 3 we define a model of action, which accommodates the interaction with the BMK; in Sections 4 and 5, we formalize CG and BMK using the EC, and describe how we tackle the conformance problem. Section 6 concludes the paper and discusses related work.

2 CG and BMK Complement Each Other

A CG is defined assuming some ideal conditions, that could not hold when applying the CG in the real medical practice. Hence, a CG cannot be interpreted as a protocol which has to be applied tout cour, and the actions prescribed by CGs cannot be interpreted as "must do" actions. The intended semantics of CGs is much more complex, and cannot be analysed in isolation w.r.t. the BMK. Informally speaking, given a patient X to which a CG $\mathcal{G}$ has to be applied in a context $\mathcal{C}$, $\mathcal{G}$ has to be interpreted as a set of *default prescriptions*: whenever X and $\mathcal{C}$ *fit* with $\mathcal{G}$'s prescriptions, they must be executed. However, X (or $\mathcal{C}$) may have peculiar features, which are not explicitly covered by $\mathcal{G}$. In such a case, the BMK must be considered to identify the correct actions. The interplay between CGs and the BMK can be very complex, as shown by the following examples.

Example 1. CG: Patients suffering from bacterial pneumonia caused by agents sensible to penicillin and to macrolid, allergic to penicillin, must be treated with macrolid.
BMK: Don't administer drugs to an allergic patient.

In Ex. 1, two alternative treatments (penicillin or macrolid) are envisaged by the CG, but one of them is excluded, given the underlying BMK, because of allergy to penicillin. Here the BMK reinforces the CG and helps to discriminate among different alternatives. In other cases, the BMK may apparently contradict the CG. However, there is no general rule in case of "apparent contradiction": in some cases the BMK recommendations "win" over CG ones, or vice versa.

Example 2. CG: Patient with acute myocardial infarction presenting with acute pulmonary edema; before performing coronary angiography it is mandatory to treat the acute heart failure.

Fig. 1. Part of the CG for acute myocardial infarction represented in GLARE

BMK: The execution of any CG may be suspended, if a problem threatening the patient's life suddenly arise. Such a problem has to be treated first.

Example 3. CG: In a patient affected by unstable angina and advanced predialytic renal failure, coronary angiography remains mandatory, even if the contrast media administration may cause a further final deterioration of the renal functions, leading the patient to dialysis.

In Example 2 the execution of a CG is suspended, due to the presence of a problem threatening the patient's life. The "contradiction" (logical inconsistency) between CG's recommendations and BMK is only apparent. It arises just in cases one interpret CG's recommendations as must do, while, as a matter of fact, they may be emended by BMK. In Example 3 instead a treatment is performed even if it may be dangerous for the patient. In some sense, not only some CG's prescriptions are "defeasible", since they may be overridden by BMK, but the same also holds for part of BMK.

When considering the conformance of an execution log w.r.t. a specific CG, additional actions not foreseen by such CG might be an issue. This could happen as a consequence of some particular routine, like in Example 4.

Example 4. Calcemia and glycemia are routinely performed in all patients admitted to the internal medicine ward of Italian hospitals, regardless of the disease.

Examples 1–4 clearly show that CGs cannot be simply interpreted as a strict, normative procedures. The context of execution and the BMK complement the prescriptions in the CGs, bridging (at least in part) the gap between the "ideal" and the "real" application cases.

Let us now better specify the Example 2, in the context of a CG for the acute myocardial infarction. The following refinement shows that both declarative and procedural knowledge usually come into play.

Example 5. CG (excerpt): Actions (Electrocardiographic study), (Echocardiographic study), and (Coronary Angiography) should be executed in sequential order. BMK: (1) *Threats to patient's life must be addressed immediately;* (2) *an acute heart failure is a life threat;* (3) *an immediate response for acute heart failure could be a (Diuretic Therapy).*

In Example 5, the knowledge of the CG is defined in terms of a procedural specification of the actions to be performed (see Fig. 1); the BMK knowledge instead is given partly in terms of desired properties and definitions (sentences (1) and (2)), and partly in terms of procedural recommendations (Sentence (3) could be read as "in case of an acute heart failure apply Diuretic Therapy").

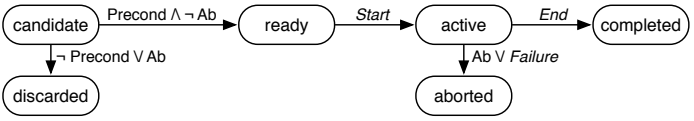

Fig. 2. The model for the execution of an action, as a transition system

Summing up, two different types of knowledge must be taken into account: the knowledge deriving from a CG, and BMK that integrates the former one. Such information is often expressed using a mixed declarative/procedural style. In this hybrid situation the property of conformance, intended as the adherence of a trace to CGs and BMK, becomes more and more important, and yet difficult to be captured.

3 The Continuos Interplay between CG and BMK

The interaction between clinical knowledge in the CG and BMK takes place during the execution of CGs. To support such an interaction, we have defined a model of the execution of actions in the CG (see Fig. 2).

At a given point in the execution of a CG on a specific patient, the control relations in the CG indicate that a given action is the next action to be executed (or, in case of parallel execution, that a set of actions is expected). At that point, we say that the action is the *candidate* (for execution) action. To be executed, a candidate action must satisfy its *preconditions*, which are a part of the description of the action itself ("precond." in Fig. 2). Preconditions specify the applicability conditions of the action, and have to be evaluated on the basis of the currently available patients data and execution context. Even in case preconditions are satisfied, the action cannot be executed if some *abnormality* ("ab" for short) situation shows up. Abnormalities arise whenever the assumptions on CG execution (*ideal* patient and context, as described in Section 1), do not hold. If the situation is not abnormal and preconditions hold, the action is *ready* to be executed. Otherwise, it becomes *discarded*. When a *ready* action is *started*, it becomes *active*. Two cases are possible then: either the active action is *ended*, leading to a *completed* action; or an *abnormality/failure* shows up during execution, so that the action is *aborted*. *Failures* denote the uncorrect completion (or no completion at all) of an action, due to human and/or technical problems arising during its execution.

It is worth stressing that the points of interactions between CG execution and BMK are explicitly modeled by the *abnormality* arcs shown in Fig. 2. The rationale is the following: whenever, during the execution of the CG, the patient/context are not *ideal* (i.e., they do not fit the assumptions made during the definition of the CG), physicians have to integrate the CG knowledge with their own abilities and expertise. In particular, they must continuously evaluate *preconditions* and *abnormality/failure* situations, then deciding how to act. Observe that the *preconditions* are specified in the CG model, the *failure* situations

Table 1. The EC ontology

$happens\_at(Ev, T)$	Event Ev happens at time T
$holds\_at(F, T)$	Fluent F holds at time T
$holds\_for(F, [T_1, T_2])$	Fluent F holds along a time interval $[T_1, T_2]$
$initially(F)$	Fluent F holds from the initial time
$initiates(Ev, F, T)$	Event Ev initiates fluent F at time T
$terminates(Ev, F, T)$	Event Ev terminates fluent F at time T
$mvi(F, T_i, T_f)$	$(T_i, T_f]$ is a *maximal validity interval* for F

depend from a specific execution, and the *abnormality* situations are typically handled by the BMK. Further constraints are imposed by the BMK depending on the current context and patient's status; from the operational point of view, such constraints might forbid or require the execution of specific actions.

4 Formalisation of the CG and BMK Using the EC

In this section we show how, in spite of their different role and knowledge representation languages, both CG and BMK can be formalized by an uniform logic framework based on the EC.

4.1 Introduction to the Event Calculus

The Event Calculus was proposed by Kowalski and Sergot [11] as a logic programming framework for representing and reasoning about time, events and their effects [11]. Basic concepts are that of *event*, happening at a point in time, and *fluent*, a dynamic property holding during time intervals. Fluents are initiated/terminated by events. Given an event narrative (a set of events), the EC theory and domain-specific axioms together ("EC axioms") define which fluents hold at each time. There are many different formulations of these axioms [5]. One possibility is given by the following axioms ec_1, ec_2 (P stands for *Fluent*, E for *Event*, and T represents time instants):

$$holds\_at(P,T) \leftarrow initiates(E,P,T_{Start}) \wedge T_{Start} < T \wedge \neg clipped(T_{Start},P,T). \quad (ec_1)$$

$$clipped(T_1,P,T_3) \leftarrow terminates(E,P,T_2) \wedge T_1 < T_2 \wedge T_2 < T_3. \quad (ec_2)$$

$$initiates(E,P,T) \leftarrow happens\_at(E,T) \wedge holds\_at(P_1,T) \wedge ... \wedge holds\_at(P_M,T). \quad (ec_3)$$

$$terminates(E,P,T) \leftarrow happens\_at(E,T) \wedge holds\_at(P_1,T) \wedge ... \wedge holds\_at(P_N,T). \quad (ec_4)$$

Axioms (ec_3, ec_4) are schemas for defining the domain-specific axioms: a certain fluent P is initiated/terminated at time T if an event E happened at the same

time, while some other fluents P_i hold. The expression $happens\_at(E,T) \wedge holds\_at(P_1,T) \wedge ... \wedge holds\_at(P_N,T)$ represents the *context* which causes E to initiate P. In general, the context can be any conjunction of literals. To say that a fluent holds at the beginning of time we can use the shorthand $initially(P)$. Note that, to maintain the reasoning consistent w.r.t. the time instants, it is usually assumed that a fluent initiated at time T holds from time T onward; a fluent terminated at time T instead still holds at time T, but it does not hold later than T. I.e., the interval time on which a fluent holds is open on the left and closed on the right. The EC formalization above is called *simple* EC and uses the Horn fragment of first order logic, augmented with negation as failure.

An EC *theory* is a knowledge base $\mathcal{KB}$ composed by a set of clauses (*initiates*, *terminates*, ...) that relate events and fluents. The set of all EC predicates that will be used throughout the paper is listed in Table 1.

4.2 Significative Events within a CG Execution

The first step to model the CG/BMK is to identify the significant events that happen in the system. Such events must be *observable*, in the sense that a system supporting the execution of a CG should be able to properly log them. Indeed GLARE[14], when supporting the execution of a CG, logs many events related to CG, as well as to the patient status and many other health-related aspects.

Among these events, we distinguish between two different types. The first type is related to the execution of actions, in particular to represent the *start*/*end* of an action's execution, as well as its *discard*/*abort*. Such events represent the state transitions presented in Fig. 2, with a slight difference. In Fig. 2 when an action is in the *candidate* status, if the preconditions hold and the current context/situation is not abnormal, then such activity becomes ready and it is executed (hence becoming *active*). This process mirrors the typical behavior of a human professional that, having the goal of executing an action, checks for the preconditions and abnormal situations, and then proceeds with the action. However, it is not reasonable to assume that the log of the CG execution will contain this kind of information: almost all the existing CG support systems log only the fact that an action has been executed. Precondition/abnormalities checks are taken for granted since the action has been executed.

The second type of events represents any other type of information that is not strictly related to the execution of an action. In this category falls events like "a patient had a heart failure at time 9" or "at time 16 the patient had a temperature of 39.7 degrees". A brief summary of the events we assume to be observable (logged) during the CG execution is given in Table 2.

4.3 The Action Execution Model in EC

In our modeling of the execution model discussed in Section 3 we use a special fluent, namely *status(A, S)*, for representing the fact that action A is in status S. Also, we assume that an action is already in the state *candidate* (the elicitation of candidate actions will be detailed in Section 4.4). In such state, our model

Table 2. Observable events

$exec(event(start, A))$	The execution of action A has been started
$exec(event(end, A))$	The execution of action A has been ended
$exec(event(discard, A))$	The candidate action A has been discarded by the operator
$exec(event(abort(R), A))$	During its execution, A has been aborted for reason R
$heart\_failure$	The patient has a heart stroke
$temperature(36.5)$	The patient's temperature has been measured to be 36.5°

foresees two possible events. The first event is the start one, which triggers the transition from *candidate* to *active*, as specified by Axioms ax_1 (the candidate status is terminated) and ax_2 (the *active* status initiates).

$$terminates(exec(event(start, A)), status(A, candidate), T). \quad (ax_1)$$

$$initiates(exec(event(start, A)), status(A, active), T) \leftarrow holds\_at(status(A, candidate), T). \quad (ax_2)$$

Note that the termination of the *active* status is not subject to any particular condition. Instead, for initiating the new status *active*, Axiom ax_2 explicitly requires that the action A is currently *candidate*.

The second possible event is a *discard*, meaning that the operator has decided to discard a candidate action. Axioms ax_3 and ax_4 capture the transition form the state *active* to the state *discard*.

$$terminates(exec(event(discard, A)), status(A, candidate), T). \quad (ax_3)$$

$$initiates(exec(event(discard, A)), status(A, discarded), T) \leftarrow holds\_at(status(A, candidate), T). \quad (ax_4)$$

The formalization of the state transitions from *active* towards *completed*/*aborted* (as consequence of events *end*/*abort*) are similar to Axioms ax_1–ax_4; we do not report the corresponding axioms for lack of space.

4.4 Formalization of a Clinical Guideline Using EC

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. Our formalization of this workflow part is a variant of [6].Differently from [6], we focus on the elicitation of candidate actions (by raising up the proper fluent). When an action is completed correctly, other action(s) become candidates, depending on what is specified by the workflow. But also in case an action is discarded or aborted, the following actions (as specified by the CG workflow) become candidates, anyway.

The rationale behind this choice 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 discarded or interrupted (aborted) for many possible reasons,

and yet the execution of the CG is brought forward. To support such cases, the workflow part of a CG must be "robust": it should support the operators in executing the whole CG, even if some actions have been discarded/aborted. To represent the candidate action(s) as prescribed by the CG, we use a fluent *nextCGCandidate*, that is continuously updated to represent the next action to be executed, according to the CG. Note also that the action itself is put on state *candidate*, and from that moment on it is treated as specified in Section 4.3.

4.5 Formalization of the Basic Medical Knowledge in EC

The EC is a framework based on first-order logic axioms. Although many implementations are available, we are currently using $\mathcal{REC}$[1] [4], a pure Prolog implementation of the EC, built on top of a Prolog interpreter written in Java. Using Prolog and EC for representing the BMK has been a quite natural choice.Note that the definition of BMK cannot be exhaustive (i.e. it is not possible to acquire all basic medical knowledge for all medical problems). Actually, portions of the whole BMK will be captured depending on the specific medical problem (i.e., depending on the CG at hand). For example, the following knowledge base represent the fact that a heart failure is a life threat, and that a diuretic therapy is a possible treatment for it. Moreover, the knowledge base specifies also that an event representing a life threat initiates an abnormality status, that is terminated by starting any treatment for the particular life threat.

$$life\_threat(heart\_failure). \qquad treatment(heart\_failure, diuretic).$$
$$initiates(exec(E), abnormality(E), \_) \leftarrow life\_threat(E).$$
$$terminates(exec(event(start, A)), abnormality(E), \_) \leftarrow treatment(E, A).$$

5 Conformance Evaluation of an Execution Log

We aim to evaluate when the execution of a CG might not be completely conformant to the CG specification. With no claim of being exhaustive, we propose some possible interesting cases. The first case happens when the CG suggests a candidate action, but the operator starts executing a different action. Axiom ax_5 captures this situation, by raising a special fluent *status(cg,nc)*, indicating the possible non-conformance.

$$initiates(exec(event(start, A)), status(cg, nc), T) \leftarrow holds\_at(status(nextCGcandidate, B), T), A \neq B. \qquad (ax_5)$$

A second case is when an action has been started, but either the preconditions did not hold, or there was an abnormal situation (Axioms ax_6 and ax_7):

$$initiates(exec(event(start, A)), status(cg, nc), T) \leftarrow holds\_at(status(A, candidate), T), \neg preconditions(A, T). \qquad (ax_6)$$

[1] `http://www.inf.unibz.it/~montali/tools.html#jREC`

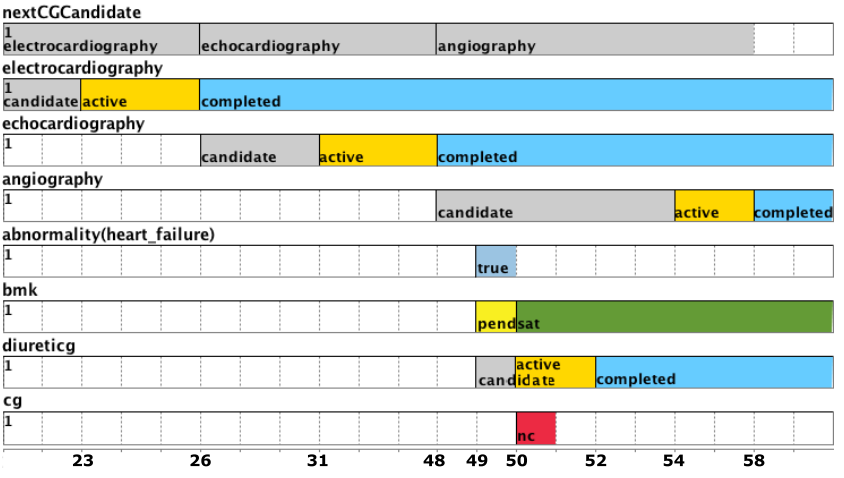

Fig. 3. EC-based conformance evaluation of a CG execution

$$initiates(exec(event(start, A)), status(cg, nc), T) \leftarrow \\ holds\_at(status(A, candidate), T), holds\_at(abnormality(\_), T). \quad (ax_7)$$

where *preconditions* is a Prolog predicate, and *abnormality* is a special fluent signaling the abnormal situation. A third, possible non-conforming situation might arise when a candidate action has been discarded, although there was no apparent reason. Such situation is captured by Axiom ax_8:

$$initiates(exec(event(discard, A)), status(cg, nc), T) \leftarrow \\ holds\_at(status(A, candidate), T), \\ preconditions(A, T), \neg holds\_at(abnormality(\_), T). \quad (ax_8)$$

5.1 A Simple Example

Let us consider the CG fragment shown in Fig. 1. Fig. 3 shows the EC-based conformance evaluation for a simple log. Initially, the candidate action is electrocardiography, that is started at time instant 23, and ends 26 (the corresponding fluent switches from candidate to active and then completed). Following the CG, the next suggested action is echocardiography, that becomes *candidate* at time as soon as the previous action is completed (time 26). The echocardiography is started at time 31 and is terminated at time 48. The next action foreseen by the CG is an angiography, that becomes candidate at time instant 48. However, at time 49 the patient has an heart failure. Such event generates an abnormal situation (signaled by the *abnormality* fluent) which triggers the BMK, activating its rules. In fact, in our example the BMK specifies that in case of heart failure, such life threat must be treated immediately, and that a possible treatment is a diuretic therapy. Thus, the action diuretic becomes candidate, and is then executed.However, from the CG viewpoint, the diuretic therapy was not expected

at all, and it raises a possible non-conformance warning, as shown by the fluent *cg*, that assumes the value "nc". Once the heart failure has been treated, it is possible to continue with the execution of the CG: the angiography, still a candidate action, is started at time 54 and completed at time instant 58.

6 Discussion and Conclusions

In this paper we focused on the interaction between clinical guidelines (CGs) and the basic medical knowledge (BMK) in the light of the conformance problem, intended as the adherence of an observed CG execution to both types of knowledge. We have provided a formalization of CGs and BMK based on EC; in particular we have defined a model of the CG action execution that accommodates the CG-BMK interaction. We aim to provide a facility to support "a posteriori" conformance evaluation: given a complete execution trace, we can evaluate whether it is conformant to the CG and BMK. Notice that we only focus on non-conformance detection, without judging whether the operatum of the physician has been correct or not. Beside supporting quality evaluation processes, our approach can be adopted for educational purposes: given a medical problem, students are called to identify the proper actions to be applied; these can be automatically compared with the ones recommended by CG+BMK.

To the best of our knowledge, many proposals (see e.g. [8,13,7,3]) have considered the BMK only as a source of definitions of clinical terms and abstractions. Instead, the BMK has been exploited in the Protocure and Protocure II EU projects, where CGs are modelled via Asbru, and the BMK is given as a set of LTL formulas. The theorem prover KIV is used to perform quality checks [10] and to check CG properties, while the "a posteriori" conformance is not addressed. Asbru semantic, based on an action model [2], shares some similarities with our semantic, but it does not consider the BMK. Another proposal that takes into account different kind of medical information is Medintel [3]: different medical information sources (e.g., guidelines, reference texts, scientific literature) are used to improve decision support and the quality of care provided by general practitioners, which can be undermined when available information is not used.

Proposals for "a posteriori" conformance have been presented in [9] and [12], respectively focusing on CGs and business processes. Both the approaches focus on the verification of the control-flow, without taking into account parameters and data associated to the actions. Moreover, Asbru provides an a-posteriori critiquing module[1] based on action intentions (i.e. goals): every actions have a set of intentions and the critiquing module checks whether the execution log coveres them. Note that in Asbru critiquing approach the BMK has not been considered.

For the sake of brevity, in this paper we have not taken into account temporal constraints in the CGs. In the future, we aim to extend our approach for run-time compliance verification. In this respect, note that $\mathcal{REC}$ has been specifically developed for run-time reasoning over execution traces, making it possible

to naturally extend the approach presented in this paper to the run-time setting. In our opinion, this will provide a significant advancement w.r.t. the other approaches to CG execution, towards integrating into the CG execution engine also the recommendations given by the BMK.

Acknowledgments. 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 Project FP7-ICT ACSI (257593).

References

1. Advani, A., Lo, K., Shahar, Y.: Intention-based critiquing of guideline-oriented medical care: The asgaard project at stanford. In: Proc. AMIA Annual Symposium, pp. 483–487 (1998)
2. Balser, M., Duelli, C., Reif, W.: Formal semantics of asbru-an overview. In: Proceedings of IDPT 2002 (2002)
3. Brandhorst, C.J., Sent, D., Stegwee, R.A., van Dijk, B.M.A.G.: Medintel: Decision support for general practitioners: A case study. In: Adlassnig, K.-P., Blobel, B., Mantas, J., Masic, I. (eds.) Proceedings of MIE 2009, pp. 688–692 (2009)
4. Chesani, F., Mello, P., Montali, M., Torroni, P.: A Logic-Based, Reactive Calculus of Events. Fundamenta Informaticae 105(1-2), 135–161 (2010)
5. Chittaro, L., Montanari, A.: Temporal representation and reasoning in artificial intelligence: Issues and approaches. AMAI 28(1-4), 47–106 (2000)
6. Cicekli, N.K., Cicekli, I.: Formalizing the specification and execution of workflows using the event calculus. Information Sciences 176(15), 2227–2267 (2006)
7. De Clercq, P.A., Blom, J.A., Hasman, A., Korsten, H.H.M.: Gaston: an architecture for the acquisition and execution of clinical guideline-application tasks. Med. Inform Internet Med. 24(4), 247–264 (2000)
8. Fridsma, D.B., (Guest ed.): Special issue on workflow management and clinical guidelines. JAMIA 22(1), 1–80 (2001)
9. Groot, P., Hommersom, A., Lucas, P.J.F., Merk, R.-J., ten Teije, A., van Harmelen, F., Serban, R.: Using model checking for critiquing based on clinical guidelines. Artificial Intelligence in Medicine 46(1), 19–36 (2009)
10. Hommersom, A., Groot, P., Lucas, P.J.F., Balser, M., Schmitt, J.: Verification of medical guidelines using background knowledge in task networks. IEEE Trans. Knowl. Data Eng. 19(6), 832–846 (2007)
11. Kowalski, R.A., Sergot, M.: A logic-based calculus of events. New Generation Computing 4(1), 67–95 (1986)
12. Rozinat, A., van der Aalst, W.M.P.: Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 163–176. Springer, Heidelberg (2006)
13. 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)
14. Terenziani, P., Montani, S., Bottrighi, A., Molino, G., Torchio, M.: Applying Artificial Intelligence to Clinical Guidelines: the GLARE Approach. In: Teije, et al. (eds.) [13], vol. 139, pp. 273–282 (July 2008)