The role of ontologies and decision frameworks in computer-interpretable guideline execution

Paulo Novais[1], Tiago Oliveira[1], Ken Satoh[2] and José and Neves[1]

[1] Algoritmi Research Centre/Department of Informatics, University of Minho
Braga, Portugal
{toliveira,pjon,jneves}@di.uminho.pt
[2] National Institute of Informatics, Sokendai University
ksatoh@nii.ac.jp

Abstract. Computer-Interpretable Guidelines (CIGs) are machine readable representations of Clinical Practice Guidelines (CPGs) that serve as the knowledge base in many knowledge-based systems oriented towards clinical decision support. Herein we disclose a comprehensive CIG representation model based on Web Ontology Language (OWL) along with its main components. Additionally, we present results revealing the expressiveness of the model regarding a selected set of CPGs. The CIG model then serves as the basis of an architecture for an execution system that is able to manage incomplete information regarding the state of a patient through Speculative Computation. The architecture allows for the generation of clinical scenarios when there is missing information for clinical parameters.

1 Introduction

Knowledge-based systems draw a clear separation of their control processes, which determine what a system should do, from their global database, which defines what a system knows [10, 35]. This separation can be translated into their basic architecture: a knowledge base where knowledge is represented explicitly rather than in procedural code, and an inference engine that runs that knowledge against information about the state of the world. These systems have a number of desirable features such as: making the information required to solve a decision problem explicit, their maintenance is faster and easier with the separation of domain knowledge and code, and the ability to explain the outputs they produce. In clinical decision support these features play an important role given the need for well-founded and consistent advice [17]. However, one of the most difficult parts in developing knowledge-based Clinical Decision Support Systems (CDSSs) is obtaining the necessary domain knowledge. Experts have a limited schedule, their attention is highly demanded throughout their daily activities, in multiple situations. Since computer-based clinical decision support is not a priority, the task of developing these systems is often overlooked in favour of tasks that are more urgent and impactful in the short term. Despite that, there are vehicles for clinical knowledge that can be used as a support for the development of CDSSs. Such is the case of Clinical Practice Guidelines (CPGs) [22, 34], which are systematically developed statements, based on medical evidence, that provide advice for specific patient

states. CPGs cover a wide variety of clinical categories and medical specialities, from diagnosis to management, from family practice to surgery. They aim to promote the standardization of medical practice as a way to prevent deviations responsible for the occurrence of adverse events or medical errors [42]. Additionally, they are an important tool in combating over-expenditure resulting from defensive medicine [37].

The versions of CPGs in machine-readable formats are called Computer-Interpretable Guidelines (CIGs) [19, 30, 26]. The advantages of adopting these machine-readable versions over their document counterparts are related with the increased availability of guidelines at the point and moment of care and reduced ambiguity [6]. CIGs also help reduce the vagueness in clinical documents, namely that which stems from the use of fuzzy terms to describe events or recommendations whose reach is difficult to determine [6]. Furthermore, the existence of structured formats allows for the development of automated mechanisms for the interpretation of clinical knowledge, resulting in knowledge-based systems that help physicians make better decisions.

Research in CIG systems started in the late 1980's and took off in the late 1990's and early 2000's [30]. Currently, there are a number of CIG models focused on different aspects of CPG representation. These aspects are related with the basic requirements for building CIG models, namely the representation of [5]:

- Workflows of recommendations, enabling the definition of sequences of recommendations, alternative recommendations, and parallel recommendations;
- Conditions about the state of the patient that restrain the application of clinical recommendations;
- Decision points for the inference of new patient states; and
- Temporal restrictions such as durations, waiting times and periodicities.

When developing a CIG model and corresponding execution engine, these are aspects that must be taken into account. They establish the foundations for the development of higher level functions in CIG execution engines [24]. One of such high level functions is the management of uncertainty, which is a pervasive problem in health care. There are multiple meanings and varieties of uncertainty in health care, which are not often distinguished [20, 1, 23, **?**]. According to [13], where a taxonomy for uncertainty in health care is proposed, this concept can be defined as the perception of not having knowledge about some aspect of the real world. The form it takes depends on many factors such as the source of uncertainty and how it manifests. The kind of uncertainty addressed throughout this chapter falls within the category of incomplete information, discussed in [20], and it can be described as the lack of knowledge about the correct values for the parameters of a model, in this case, about the data regarding the state of a patient, necessary for CIG execution. Given this initial presentation, the objectives of this book chapter are the following:

- To highlight the advantages of using an already established ontology language to develop a CIG representation model;
- To demonstrate how the procedural and domain knowledge of a CPG can be represented in an ontology and to enumerate the necessary representation primitives to do so;

– Showcase an architecture for a CIG execution environment based on the developed ontology that not only provides the necessary elements for CIG execution, but also a module for the management of incomplete information;
– Propose a method for handling incomplete information regarding data entry points in CIGs in order to produce clinical scenarios of guideline executions.

In order to fulfil the above-mentioned objectives, the chapter is organized as follows. In Section 2 we briefly present the current state of the art in CIG research and identify limitations that may pose as research opportunities. We describe a CIG representation model based on Web Ontology Language (OWL) in Section 3. In this section we also specify how the model fulfils the basic requirements for the representation of CIGs. Additionally we show and discuss the results of an evaluation performed on the expressiveness of the model. In Section 4 we propose a CIG architecture for the execution of guidelines and management of incomplete information. We describe each element and focus mainly on a module based on speculative computation. Finally, in Section 5 we draw conclusions from the presented work and make considerations about development perspectives.

2 Modelling and Executing Computer-Interpretable Guidelines

The initial step towards the modelling of clinical recommendations from CPGs in CDSSs was the development of the HELP system [12]. Despite its success, the development of CDSSs based on CIG models only continued in the late 1980s. Arden Syntax [31] was one of the early approaches to CIG modelling. It represents CPGs as sets of independent modules, each one called a *medical logical module* (MLM). A MLM has a structure organized in layers. The first is the *knowledge layer* and contains the clinical knowledge in the form of if-then-else rules whose premises are related to the state of the patient. The *administrative information layer* provides information such as authoring and purpose of the guideline. As for information regarding updates and version, they are kept under the *maintenance layer*. This type of representation handles only a single decision in a CPG and it views guidelines as independent building blocks in the clinical process. However, in reality, the relationships between these knowledge blocks is considerably more complex than this. As such, the most substantial limitation of Arden Syntax is that it is not the most appropriate format for developing complete electronic guideline applications. This model is one of the standards from Health Level 7 (HL7) for medical knowledge.

Another approach to CIG modelling is the Guideline Interchange Format (GLIF3) [29, 3]. It is an approach that organizes CPGs in *steps*, the basic building blocks of the model. The different types of steps are *decision*, *patient state*, *action*, *synchronization* and *branch*. A decision step encodes a decision moment in the CIG workflow, it is used to infer new information based on the state of the patient. The enactment of clinical procedures is encoded in *action* steps, whereas the retrieval of patient information is performed through *patient state* steps. The branching of clinical pathways and their further synchronization is achieved with the *branch* and *synchronization* steps respectively. GLIF3 is an extended formalism that emphasizes the sharing of CIGs across

institutions and focuses heavily on the integration of medical knowledge with medical terminologies. However, it relies on subsets of other CIG languages in order to express elements such as temporal constraints on the execution of steps, which may be viewed as a disadvantage.

Temporal constraints are the main focus of the Asbru [33, 32] model for CIGs. Clinical recommendations are represented as decomposable plans with complex temporal patterns and annotations. A plan is a collection of items called *knowledge roles*, which can assume different forms, namely: *preferences*, *plan intentions*, *conditions*, and *effects*. These items define which actions to perform, what is intended with them, what conditions should be gathered to perform them, and what is expected to happen after they are applied. Additionally, Asbru defines a comprehensive set of temporal constructs to represent time. These temporal constructs include constraints on the starting time and ending time of tasks (such as earliest possible start and earliest possible ending), maximal and minimal durations, and cyclical time points (e.g., every morning, every day, etc.). The downside to this is some criticism regarding the complexity of temporal annotations, which seem to be unable to match the knowledge elements in CPGs, particularly those that concern cyclical executions of procedures.

The GuideLine Acquisition, Representation and Execution (GLARE) [36] is a model consisting of atomic and composite *actions*. The *action* is its basic unit, featuring different types such as *work action*, *query action*, *decision* and *conclusion*. The basic building blocks of GLARE follow the structure and meaning of the other models that were already presented. The difference here is the *conclusion* action, which represents the explicit output of a process. GLARE has a comprehensive set of temporal constructs, especially when it comes to cyclical tasks. However, this model is limited in the representation of temporal constraints involving the evolution of the state of a patient.

The Standards Active Guideline Environment (SAGE) [38] is a CIG project that aims to establish a framework for acquiring and sharing guidelines in multiple health care systems. SAGE applies the EON model for the machine-readable formalization of CPGs. Within the model, a CPG consists of *recommendation sets*, represented as a graph of context nodes. Each node is an instance of one of three classes: *action*, *decision* and *routing*. SAGE uses standard terminologies, such as SNOMED-CT [7] and LOINC [9], to provide unequivocal meaning to clinical terms.

All of the above-mentioned modelling approaches have hardly transitioned from the research phase to wide real world implementations, with the most prominent exceptions being Arden Syntax and SAGE. In [28] it is possible to consult an inclusive summary of CIG usage and applications. Moreover, most CIG models require proficiency in programming languages to express logical rules or temporal constraints, which is impractical for health care professionals. With this description of CIG models we intend to show different modelling perspectives, driven by different goals. Whether the focus is placed on decision points, the accurate representation of workflows of procedures, or temporal constraints, there are common features to all models, which we show in the next section. There are other relevant approaches such as PROforma [11] or GUIDE [4]. For an insight on these models and a more detailed overview of CIGs, we urge the reader to consult the works in [26] and [30].

In order to properly run CIGs against patient data and obtain clinical recommendations, it is necessary to develop an algorithm, often referred to as execution engine, that analyses the CIG elements given the context provided by the state of the patient. Surrounding the execution engine, there should be a proper system that manages all the workflow from user inputs to the production of recommendations and automatic inferences. Such a system should facilitate the inclusion of CIG advice in different settings [8]. Guideline execution engines such as GLEE [41] for GLIF3, the Spock Engine [43] for Asbru or the GLARE Execution Engine [36] were specifically developed for running CIGs in health care settings. Most of them, including the mentioned examples, exist in the form of client-server applications, with the execution engine placed on the client side. Furthermore, these applications are mostly available as desktop applications, which is an obstacle to their reach and ease of deployment. An extensive review on the computer-based execution of clinical guidelines can be found in [15]. These execution engines could take a better advantage of their knowledge base, i.e, the way in which CIGs are represented and of their knowledge elements, as well as of their patient case base in order to address situations outside of the constraints defined in the guideline workflow. A common example of this is the existence of missing or incomplete information for a data entry point in a guideline, which renders the execution engine unable to produce a clinical recommendation.

3 A CIG model based on an ontology language

Web Ontology Language (OWL) is an ontology standard developed by the World Wide Web Consortium (W3C) [21]. The second version of OWL (OWL 2) has increased expressiveness and it is built upon formats such as XML, RDF and RDF-schema. The description logics version of OWL (OWL DL) provides additional vocabulary and its components allow for an easy and expressive representation of the knowledge elements in CIGs. The components that make this possible are:

- Classes: sets that contain individuals described using formal (mathematical) descriptions that state precisely the requirements for membership of the class;
- Individuals: objects of the domain and instances of classes; and
- Properties: binary relations on individuals that may be used to link two individuals (object properties) or an individual to a data element (data properties).

The advantages of this ontology language in the representation of CIGs are related with the internal structure of an ontology in OWL DL. These components are organized in a graph database that is unlike the more common relational and hierarchical databases (nodes and tables). This makes the connection between knowledge components easier and clear. The relationships in OWL assume a greater importance and are the carriers of the semantic content of individuals. Moreover, it is possible to describe or restrain class membership using these relationships and thus accurately delimit their scope.

There are essentially two ways of developing a CIG model. One is consulting domain experts in order to specify the representation primitives. The other is researching different CPGs and determine the information needs of clinical recommendations. The method followed in this work was hybrid in the sense that it included opinions from

a health care professional and the observation of guidelines collected from the National Guideline Clearinghouse (NGC)[3]. The developed ontology for CPGs was called CompGuide. In it, complex information elements are represented as individuals with multiple object properties connecting them to other individuals, and simple information elements that cannot be further decomposed are represented using data properties. However, simple information that is reusable and will most likely be needed across different points of a CPG is represented as class individuals as well. In this regard the representation is similar to a linked list of procedures. As such, a CPG is represented as an instance of the *ClinicalPracticeGuideline* class. Individuals from this class have a set of data and object properties that enable the representation of descriptive and administrative guideline information such as the name of the guideline, its general description, date of creation, date of last update, version, clinical speciality, category, intended users, and target population.

The CompGuide model was initially presented in [25], where a more detailed description of the model can be found. In the following sections we will present the CompGuide CIG model under the basic requirements for building CIG models defined in Section 1.

3.1 Definition of workflows of recommendations

Like other CIG models, CompGuide follows a task network model (TNM) in which all the knowledge elements of CPGs are represented as different tasks. The classes that enable this are:

- *Plan*: a composite task that hosts any number of instances of other tasks, including other plans. An individual of *ClinicalPracticeGuideline* representing a CPG has exactly one global *Plan* within which are hosted all of its tasks. A *Plan* is connected to its first task with the *hasFirstTask* object property;
- *Action*: an activity that should be carried out by a health care professional. It is possible to define subtypes of *Action* via an object property connecting the individual of *Action* to an individual belonging to *Exam*, *MedicationRecommendation*, *NonMedicationRecommendation* and *Procedure*, which specify different types of actions;
- *Question*: a task to obtain information about the health condition of a patient, more specifically about the clinical parameters necessary to follow the guideline. The source of this information can be a human input or an existing database. This class is associated with data properties to define the clinical parameter to be retrieved and the units in which it should be expressed;
- *Decision*: a task representing an inference made in the clinical workflow regarding the state of the patient, based on a set of clinical parameters that act as premises. A common example of this situation would be a diagnosis.

These different classes of tasks, along with the classes used to encode conditions regarding the state of a patient, enclose the domain knowledge of a CPG. The procedural knowledge is defined by the connections that exist between the individuals of these

[3] Available at https://www.guideline.gov/

classes. In order to connect individuals belonging to the classes of tasks there is a set of object properties that establish the relative order between them. The definition of sequential tasks is possible with the *nextTask* property. For cases in which, at a splitting point in the clinical workflow, it is necessary to execute one of multiple alternative tasks, the current task is connected to the alternatives with the *hasAlternativeTask* object property. Another situation is when there is a set of tasks that should be executed simultaneously. In this case, the current task is connected to the following tasks with the *hasParallelTask* object property. For both alternative tasks and parallel tasks there are synchronization tasks where the workflow converges to a single execution path.

Although this work draws some inspiration from pre-existing models, such as Arden Syntax, PROforma, GLIF3, Asbru or SAGE, it also introduces different views on the definition of a clinical workflow using the native elements of an ontology language, in this case of OWL.

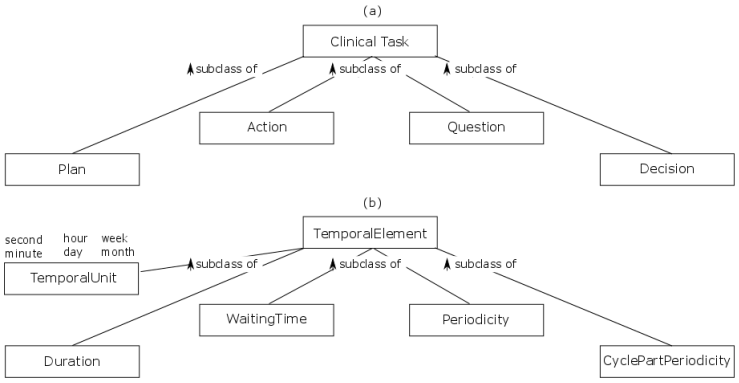

Fig. 1. Main classes for the representation of (a) clinical tasks and (b) temporal patterns.

3.2 Conditions about the state of a patient and decision points

In CompGuide, conditions about the state of a patient are associated with the tasks at which they are verified. In this regard, there are three types of conditions represented by the following classes:

- *TriggerCondition*: this is a condition regarding the clinical parameters of a patient that is used to select an alternative task in the clinical workflow. An alternative task, such as the ones mentioned in Section 3.1, has associated trigger conditions, which, when validated, dictate the selection of the task. Trigger conditions can be defined for any type of task;
- *PreCondition*:this condition is used for all types of task to express the requirements of the patient state that must be met before the execution. For instance, when administering some pharmacological agent it should be known that the patient is not allergic to it;

– *Outcome*: this condition puts a restriction to a *Plan* or an *Action* representing the result in terms of the evolution of the state of a patient to be achieved with the task. A typical use for this *Condition* is the specification of therapy goals in an *Action*.

Each of the above-mentioned conditions is connected to an individual of the class *Condition*, which actually allows for the specification of the clinical parameter in question and the value (or range of values) that fulfil the constraint, along with the units in which the parameter is expressed. This class is also important in the definition of the *Decision* class, which consists of a set of individuals of the *Option* class, each one connected to an individual of the *Condition* class. An individual of *Option* specifies a possible conclusion for the *Decision* task, and, in turn it is connected to the *Condition* that supports the option.

Given this exposition about the types of constraints placed on tasks, it is possible to identify four different types of decision points in CompGuide, namely: i) the selection of an option in a *Decision* task, ii) the selection of an alternative task based on a *TriggerCondition*, iii) determining whether a task should be executed or not based on the verification of a *PreCondition*, and iv) determining if a task was effective based on the verification of an *Outcome*.

3.3 Temporal constraints on the execution of tasks

Time is a crucial dimension in the representation of clinical procedures. This is denoted by the number of CIG representation models that are temporally oriented [33, 32, 36]. The temporal constraints in CPGs are used to express a variety of elements that need to be controlled in order to ensure the correct application of recommendations and the proper management of patients. In [27] the temporal aspects of CompGuide are explored and the main classes for temporal representation are described. These classes aim to represent the patterns featured in clinical procedures, namely durations, repetitions, periodicities and waiting times. This representation is achieved with subclasses of *TemporalElement* shown in Figure 1 (b). The meaning of each one is the following:

– *Duration*: an individual of the class *Duration* allows for the specification of how long an *Action* or a *Plan* should last, since these are the only tasks that may unfold over time. The object property *hasDuration* connects the tasks to the respective *Duration*. This *Duration* can have an exact value or be defined with maximal and minimal values;
– *WaitingTime*: this class stands for a delay in a task used, for instance, to observe an effect on a patient of a previous task. This pattern can be used in any type of task with the *hasWaitingTime* object property;
– *Periodicity*: the class is used to define a cycle of execution for any task. It is possible to define the frequency with which the task is executed and a duration (through the reuse of the *Duration* class) to specify for how long the cycle should last. Alternatively it is also possible to specify the number of repetitions or a stop condition for the task. An individual of *Periodicity* is connected to a task through the *hasPeriodicity* object property;

– *CyclePartPeriodicity*: this class represents a temporal pattern in which there is a nested periodicity, i.e., each cycle of the cyclical task has, itself, a periodicity. Has such, whenever needed, an individual of *CyclePartPeriodicity* is connected to an individual of Periodicity with the *hasCyclePartPeriodicity* object property.

Each temporal pattern, which is the same to say each class, has an associated temporal unit. This is achieved with *TemporalUnit*, an enumerated class that consists of the individuals *second*, *minute*, *hour*, *day*, *week*, *month*, and *year*.

This model offers a comprehensive representation of temporal patterns, at the level of GLARE and more complete than models such as Asbru, GLIF3 and PROforma [27]. At execution time, the CIG execution engine builds a map of guideline execution for the tasks that have temporal constraints and, then, it performs a series of verifications on the actual starting and ending times of each one.

3.4 Expressiveness of the representation model

In order to assess the expressiveness of the ontology elements presented in Sections 3.1, 3.2, and 3.3 a study was conducted with 14 students from the fourth year of the Integrated Masters in Biomedical Engineering, branch in Medical Informatics, from the University of Minho, in Portugal. The students were familiar with both the CompGuide ontology and the Protégé ontology editor. The study consisted in asking the students to represent a set of CPGs extracted from the NGC. Each student had to represent one CPG using the ontology. After the assignment they were asked to provide feedback in the form of answers to a questionnaire and short comments regarding the expressiveness of the ontology, namely on whether it was possible to completely represent the CPGs using it. The list of CPGs used in the study is showed in Table 1. A much as possible, there was an effort to include CPGs with multiple categories, namely diagnosis, evaluation, treatment, and management. The statements used in the questionnaire complete the general statement: "The CompGuide ontology allowed the representation of:". The answers were provided in a five point Likert rating scale [16] (*1-strongly disagree*, *2-disagree*, *3-neutral*, *4-agree*, *5-strongly agree*). The statements can be seen in Figure 2, along with the results. The diverging stacked bar presents the total percentage of agreement (calculated as *agree* + *strongly agree*), the total percentage of disagreement (calculated as *disagree* + *strongly disagree*), and the percentage of participants who were neutral (equal to the percentage of the *neutral* category), for each statement.

The statements are related with the basic requirements defined for the representation of CIGs. Items 1-9 can be placed in the definition of workflows of recommendations. Here the level of agreement was at least equal to or above 50%. Indeed, the item about medication prescriptions (item 1) is the one that has the lowest agreement, the highest percentage in the *neutral* category (43%), and the only one that has percentage in the *strongly disagree* category (7%), which is indicative that the corresponding representation primitive for this action may not address all the cases featured in the CPGs. Despite this, both items 2 and 3, which correspond to the representation of other types of actions, seem to correspond to the requirements of guideline representation as they have high percentages of agreement. However, item 4, directly related with the definition of a

Table 1. List of the guidelines that were used in the study, featuring their name, organization and the number of people assigned to their representation.

Clinical Practice Guideline	Organization	People Assigned
Clinical Practice Guidelines in Oncology - Colon Cancer	National Comprehensive Cancer Network	2
Clinical Practice Guidelines in Oncology - Rectal Cancer	National Comprehensive Cancer Network	2
Clinical Pratice Guidelines in Oncology - Distress	National Comprehensive Cancer Network	2
Clinical Practice Guidelines in Oncology - Palliative Care	National Comprehensive Cancer Network	2
Detection,Evaluation,and Treatment of High Blood Cholesterol in Adults	National Heart Lung and Blood Institute	1
Diagnosing and Managing Asthma	National Heart Lung and Blood Institute	1
Diagnosis, Evaluation and Management of von Willebrand Disease	National Heart Lung and Blood Institute	1
Diagnosis and Treatment of Respiratory Illness in Children and Adults	Institute for Clinical Systems Improvement	1
Diagnosis and Management of Diabetes	Institute for Clinical Systems Improvement	1
Diagnosis and Treatment of Ischemic Stroke	Institute for Clinical Systems Improvement	1

Question is the one that has the highest percentage of disagreement (14%). In the comments provided along with the questionnaire it was mentioned that the *Question* lacked a data property where it would be possible to provide an extended detailed description of the clinical parameters that the task aims to obtain, besides the actual definition of the parameter and units. It is possible to consider that the ontology allows the representation of series of tasks and its internal organization mimics that of the clinical workflows in CPGs. This is evident in the high levels of agreement of items 5-9. Overall, the organization of the procedural logic of the guideline and the grouping of tasks in plans was considered to be advantageous, mainly because this helps the delimitation of different diagnoses and treatments. The item that refers to this grouping of tasks, item 6, has an agreement of 100%. Despite this, in the submitted comments concerns were expressed regarding the range of the available subtypes of actions. CPGs also have knowledge encoded as index tables in order to calculate health indexes that are later used in decision making. This type of knowledge could not be represented, which is an aspect to improve. A positive feedback was that, by following the design pattern of the ontology, the participants were able to find redundant elements in the guideline workflows, which did not trigger any kind of event or have any consequences further ahead in the clinical process.

In terms of conditions about the state of a patient, the levels of agreement of items 10, 11, and 12, referring to trigger conditions, pre-conditions, and outcomes was fairly high, which means that these constraints fulfilled, for the most part, their role. The items referring to decision points (items 5, 10, 11, 12) all had high agreement rates, which indicates that they are sufficiently expressive to model decision making in CPGs.

In the selected guidelines there were no cases of complex temporal patterns. In fact the most common patterns were durations and periodicities. As such, the questionnaire only covers these two temporal elements, along with repetitions, which were also present. The items referring to temporal restrictions, namely items 13-15, have low agreement when compared to the majority of the other items in the chart. Actually, they are among the items that have the highest percentage in the neutral category. It is possible to understand this through the comments of the participants, in which it was pointed out that, although it was possible to represent the temporal patterns in the CPGs, it was necessary to adapt the statements in the guidelines to fit the available constructs.

Given the difficulty and time-consuming nature of the task proposed to the participants, it would be impractical to repeat the study in a larger scale and have access to an

entire statistical population of interest. Be as it may, the study provided hints as to what improvements should be made, namely in: the representation of medication prescriptions, the tasks used to retrieve information from the patient, the diversity of actions offered by the ontology, and temporal constraints as a whole. Overall, it is possible to consider that the guidelines used in the survey were accurately represented in the ontology, despite the need for certain adaptations.

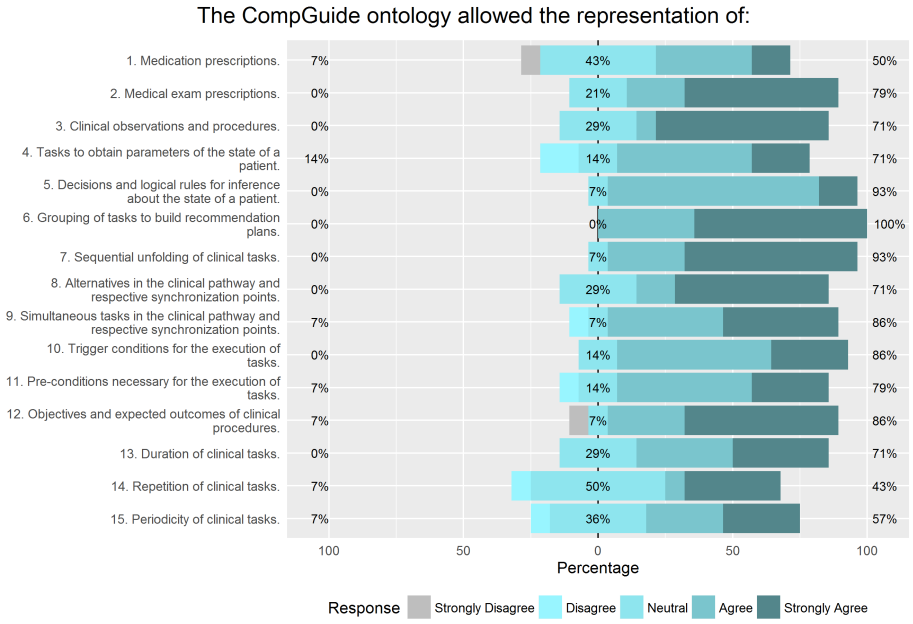

Fig. 2. Diverging stacked bar chart showing the results of the questionnaire to assess the expressiveness of the CompGuide ontology.

4 A CIG architecture for the execution of guidelines and management of incomplete information

As previously stated, a CIG-based CDSS is a knowledge-based system that uses machine-readable versions of CPGs to provide clinical recommendations. As such, the basic elements in the architecture of these systems are a knowledge base containing the CPG recommendations and an execution engine that interprets them. This is a common setting among CIG systems [15]. However, from our point of view, CIG systems should take more advantage of the expressiveness of their CIG models in order to provide additional functionalities that help health care professionals. The architecture that we

present herein aims to provide such a functionality, namely one that addresses the problem of incomplete information in decision points such as the ones presented in Section 3.2. This problem may arise due to delays in complementary means of diagnosis or the simple impossibility to know the information regarding the clinical parameters. In the following sections we present a description of the elements in the architecture and provide details regarding the knowledge flow throughout the components. Additionally, we present a *speculative module* in charge of managing incomplete information.

4.1 Elements of the Architecture

The proposed architecture can be seen in Figure 3. Its components are the following:

- *CIG repository*: this component is the knowledge base of the system. It contains *owl* files, each one representing a different CPG.
- *CIG engine*: this component is responsible for interpreting clinical guideline instructions contained in an *owl* file. It identifies different execution situations when analysing the knowledge in the CPG such as the identification of the next task from the control structures in place (sequential tasks, alternative tasks, and parallel tasks), the need to retrieve information from an external source (through *Question* tasks), and the use of that information for automatic inference in decision points;
- *local repository*: a database containing information of other patients retrieved for the data entry points in a CPG.

The problem that the *CIG engine* has to solve is the choice of the next clinical task and infer new information about the state of the patient. The information it uses is obtained from external information sources, which can be health care information systems or simply human agents providing inputs. When choosing the next clinical task from a set of alternatives based on their trigger conditions, when verifying pre-conditions before recommending a task or checking the outcome of an *Action* or *Plan*, such information sources may not be able to provide the necessary information, rendering the *CIG engine* unable to produce a decision. In these cases, a *speculative module* takes action and compensates for these information gaps. It is based on Speculative Computation with Default Revision (SCDR) [27], a logic programming theory that uses default reasoning. As such, this module intervenes when there is a *Decision* task, in the selection of an alternative task based on a *TriggerCondition*, in determining whether a task should be executed or not based on the verification of a *PreCondition*, and in determining if a task was effective based on the verification of an *Outcome*.

4.2 Speculative Module

The *speculative module* has two components. The first is the *generation of defaults* and the second is *speculative computation*. The generation of defaults assumes a supportive role and its function is to produce default values to fill in missing information regarding clinical parameters used in decision points. The *speculative module* then takes these default values and produces clinical scenarios in the form of tentative clinical recommendations and tentative inferences regarding the state of a patient. The speculative module

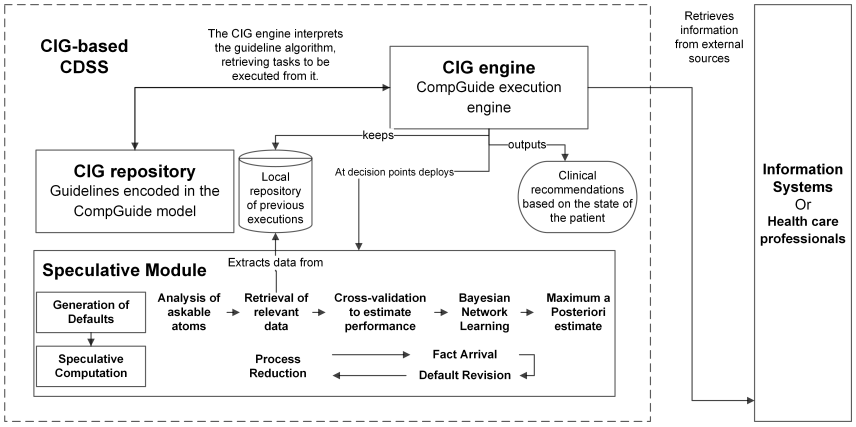

Fig. 3. Architecture for the execution of CIGs and management of incomplete information.

does not ignore the general method for dealing with uncertainty in health care, which is to use past experiences in order to fill in the missing pieces, but it is, instead, a form of using these past experiences in a more flexible way, fitter for CIG-based CDSSs than, for instance, Case-based Reasoning (CBR). The reason for this assumption is that the speculative computation used in the module offers mechanisms to manage information that are not as rigid as the complete CBR cycle. In the work [14] featuring speculative computation, fixed default beliefs are used in speculative computation. However, when applied to a real setting, the default beliefs are highly dependent on the context. The same is to say they depend on the set of circumstances and facts that surround a problem and change over time.

Generation of defaults The *generation of defaults* is a set of procedures that seek to acquire the most likely values for the clinical parameters involved in decision points, based on past executions of the same CPG for other patients. In order to take into account possible dependence relationships between the clinical parameters of a decision point, there has to be a default model capable of conveying these relationships in a direct and straightforward way. Bayesian Networks (BNs), for their set of characteristics [40, 39], provide an ideal support for such a task.

The *generation of defaults* is depicted in Figure 3. It comprises five sequential procedures. The first is the identification of askable atoms, in which the clinical parameters for the decision are identified and isolated. This requires the analysis of the individuals of *Condition* attached to an individual of *Decision*, *Trigger Condition*, *PreCondition*, or *Outcome* in order to extract the parameters whose values are the premises to infer a patient state or a new task in the workflow. Next, the module retrieves relevant data about previous guideline executions regarding the isolated parameters from the local repository. In the following procedure, different BN learning algorithms are used in cross-validation. The objective is to select the one that best conveys the relationships between the parameters. It is possible to do this with a measure of the likelihood of data

given the produced model such as the log likelihood loss (*logl*). After the best performing algorithm is selected, a BN is generated. Through a maximum a posteriori estimate (MAP) it is possible to provide the most likely values for the parameters, given the evidence, along with a probability value for the whole distribution [18]. If no evidence is known, i.e., if no value for the set of clinical parameters is available, a MAP can be submitted without evidence, in which case a set of default values is generated for each clinical parameter.

The advantage of using BNs and the MAP estimate to produce defaults is that it is possible to adjust the default values throughout the computation of a clinical recommendation. In fact, that is the principle of SCDR. At the beginning of the decision making process, when no information is available, it is possible to use the BN to generate defaults for all the parameters. However, if suddenly information arrives from the external information sources, it is possible to recalculate the default values by submitting a new MAP query, this time with the value of the known clinical parameter as evidence, thus obtaining default values for the remaining unknown parameters that depend on the piece of information that is actually known at the moment.

Speculative computation SCDR acts as the decision framework for the decision points in a CPG. The elements defined in this framework and its operational semantics enable the management of situations of completely unknown information, partially known information and completely known information. The framework is defined as the tuple $\langle \Sigma, \mathscr{E}, \Delta, \mathscr{P} \rangle$, where [27]:

- Σ is a finite set of constants, each one representing an external information source responsible for providing information on clinical parameters;
- $\mathscr{E}$ is a set containing the decision criteria, i.e., the clinical parameters used as premises in the decision points;
- Δ is the default answer set, consisting of default values for the clinical parameters in $\mathscr{E}$, obtained from the generation of defaults;
- $\mathscr{P}$ is a logic program of the form: $H \leftarrow C \parallel B_1, B_2, \ldots, B_n.$, where H is a positive ordinary literal called a head of rule R; C is a set of constraints; and each $B_1, B_2, \ldots, B_n$ is an ordinary literal, or an askable literal. $\mathscr{P}$ results from the mapping of the procedural logic and domain knowledge of the decision point leading to the recommendation of a clinical task or to the inference of a patient state.

SCDR starts with a top goal. The notion of goal is central for it represents what is necessary to achieve in the computation, or, in other words, it is the outcome of the decision. The initial set of beliefs about the state of the patient that the framework uses, if there are no known values, is Δ. Goals and the product of, their reduction, i.e., their matching with rules in $\mathscr{P}$, are kept in processes. They are structures that represent the different alternative computations in the framework. In this regard, there are two types of processes: active and suspended. Active processes are those whose constraints are consistent with the current set of beliefs of the framework. They are regarded as the valid clinical scenarios. Processes that do not fulfil this condition are suspended.

SCDR has a total of three phases: the *process reduction* phase, the *fact arrival* phase, and the *default revision* phase.

Process reduction corresponds to the normal reduction of goals within active processes. It implies the matching of goals with the head of rules in $\mathscr{P}$ and their replacement in the process with the body of the rules they are matched with. If the goal corresponds to one of the clinical parameters, they are reduced with the belief that the framework has about the parameter. If the process is consistent with that, it remains active, otherwise it is suspended. This belief can be either a default or a known value. *Process reduction* continues until an active process with an empty goal is obtained, in which case it is possible to advance a conclusion as an answer to the initial query. If the process is obtained based on default values, it is considered a clinical scenario.

In the *fact arrival* phase there is information about the real value of a clinical parameter that arrives at the *CIG engine* and is passed on to the *speculative module*. This information is handled in the following way. The beliefs of the framework are updated and the default value for the clinical parameter is replaced with the real value. It can be the case that the real value is the same as the default. As a result, all the processes, both active and suspended, are revised. Those that are consistent with the real value stay or become active and those that are inconsistent are discarded because the new information is considered to bee definitive and, thus, cannot possibly be revised again. Following this phase there is always a round of *process reduction*.

The *default revision* phase results from the verification of changes that the newly arrived information may cause in the default values of the other clinical parameters. As such, a new MAP query is submitted having the known information as evidence. If the retrieved values for the remaining clinical parameters are different from the previous defaults, then they replace the old defaults in the beliefs of the framework. Both the active and the suspended processes are revised and only the processes that are consistent with the new defaults remain or become active, the rest is suspended. This *default revision* phase ensures that the active processes move progressively to the actual true recommendation, since the framework is able to respond to the arrival of information by adjusting the other defaults to values that are closer to the respective real values, according to the probability distribution of the underlying BN. After default revision, another round of process reduction occurs.

4.3 Generation of clinical scenarios

Although in the presented architecture there is a clear separation between the knowledge base of the system and its execution engine, the design of the CompGuide ontology determines to a great extent the procedures of the execution engine. The ontology defines a set of decision points that are identified by the *CIG engine* and then mapped to the SCDR framework, which structures the reasoning process and endows the system with dynamic belief revision capabilities. This is seen when the active processes, standing for clinical scenarios, resulting from the different phases of SCDR are changed and updated into new active processes and, thus, new scenarios. In situations where the results of clinical exams may take some time to be known or may turn out to be inconclusive the effect of speculative computation yields tentative answers that enhance the capacity of health care professionals to make decisions.

If one considers the complexity of a guideline such as the Clinical Practice Guidelines in Oncology Colon Cancer [2] (which was one of the CPGs used in the study), with

multiple data entry points in the form of individuals of *Question*, followed by decision points consisting in the choice of alternative tasks (such as the clinical workflow showed in Figure 4), it is possible to use the *speculative module* on each one and, through it, present the most likely execution threads by summing the computation of these choices. Figure 4 shows an example of how this could be applied to a clinical workflow represented in CompGuide. For every decision point in the algorithm the *speculative module* runs on top of the procedural knowledge provided by the ontology. Assuming that information is missing in each *Question* task, the *speculative module* formulates a probable choice for the next task at Question1, Question2, and Question3. Then, by grouping the proposals, it is possible to build a tentative execution path, which is shown by the dashed lines in the figure. This would be useful for a practitioner as it would provide him a map of the potential evolution of a patient, thus giving him time to devise countermeasures if it shows that the treatment is following an undesirable direction.

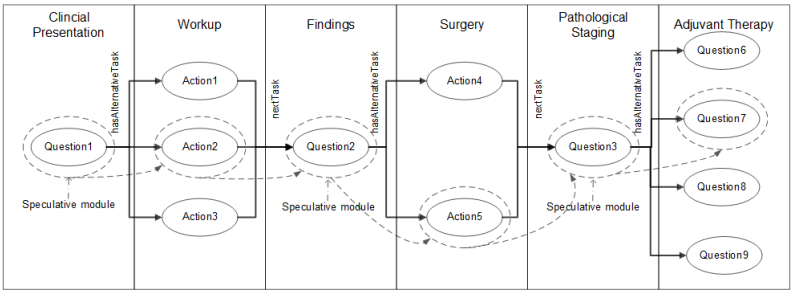

Fig. 4. Stages of the NCCN Guideline for Colon Cancer showing a symbolic representation of tasks as questions and actions throughout the clinical process: clinical presentation, workup, findings, surgery, pathological staging and adjuvant therapy. The dashed trace shows an execution path obtained from the *speculative module*.

5 Conclusions and development perspectives

It was established that the CompGuide representation model was able to provide enough expressiveness for a set of CPGs that included multiple categories and specialties, in terms of the basic requirements for a CIG model. The CompGuide ontology enables the creation of modular knowledge components and thus their reuse in different points of a CPG. By analysing and identifying the decision points in a CPG represented according to CompGuide, it is possible to map the procedural and domain knowledge of these points into a speculative computation framework that manages incomplete information. This is the basis of a *speculative module*, responsible for building clinical scenarios, based on default values retrieved from BNs. The use of probabilities, and more specifically BNs, is motivated by the notion that the knowledge we have about the world is imperfect and that, through a Bayesian approach, it is possible to get a degree

of belief that something may be the case. The inclusion of the *speculative module* is a differentiating factor from other CIG execution engines such as GLARE, GLEE and SAGE [15], which only execute their encoded rules, without additional functionalities.

There are, however, aspects to improve in the ontology in terms of knowledge representation, namely in the definition of action tasks and temporal constraints. Additionally, the utility of the clinical scenarios is not considered in the SCDR framework, but it is an important dimension since it useful for the health care professionals to know how reliable a scenario is. As such, the extension of the framework to accommodate the computation of utilities based on the probabilities provided by the BN model is a development perspective that we will follow.

Acknowledgements

This work has been supported by COMPETE: POCI-01-0145-FEDER-0070 43 and FCT Fundação para a Ciência e Tecnologia within the Project Scope UID/CEC/00319/2013. The work of Tiago Oliveira is supported by a FCT grant with the reference SFRH/BD/85291/2012. This work was partially developed during an internship program of the National Institute of Informatics (NII) of Japan by Tiago Oliveira.

References

1. Babrow, A., Kasch, C., Ford, L.: The Many Meanings of Uncertainty in Illness: Toward a Systematic Accounting. Health Communication 10(1), 1–23 (1998)
2. Benson, A., Bekaii-Saab, T., Chan, E., Chen, Y.J., Choti, M., Cooper, H., Engstrom, P.: NCCN Clinical Practice Guideline in Oncology Rectal Cancer. Tech. rep., National Comprehensive Cancer Network (2013), http://www.nccn.org/professionals/physician{_}gls/f{_}guidelines.asp
3. Boxwala, A.A., Peleg, M., Tu, S., Ogunyemi, O., Zeng, Q.T., Wang, D., Patel, V.L., Greenes, R.A., Shortliffe, E.H.: GLIF3: A representation format for sharable computer-interpretable clinical practice guidelines. Journal of Biomedical Informatics 37(3), 147–161 (2004)
4. Ciccarese, P., Kumar, A., Quaglini, S.: NEW-GUIDE: a new approach to representing clinical practice guidelines. Advances in Clinical Knowledge Management (Figure 1), 15–18 (2002)
5. de Clercq, P.A., Blom, J.A., Korsten, H.H.M., Hasman, A.: Approaches for creating computer-interpretable guidelines that facilitate decision support. Artificial intelligence in medicine 31(1), 1–27 (2004)
6. Codish, S., Shiffman, R.N.: A model of ambiguity and vagueness in clinical practice guideline recommendations. In: AMIA Annual Symposium Proceedings. vol. 2005, p. 146 (2005)
7. Cornet, R., Schulz, S.: Relationship groups in SNOMED CT. Studies in Health Technology and Informatics 150(0), 223–227 (2009)
8. Costa, R., Neves, J., Novais, P., Machado, J., Lima, L., Alberto, C.: Intelligent Mixed Reality for the Creation of Ambient Assisted Living, pp. 323–331. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
9. Dugas, M., Thun, S., Frankewitsch, T., Heitmann, K.U.: LOINC(R) Codes for Hospital Information Systems Documents: A Case Study. Journal of the American Medical Informatics Association 16(3), 400–403 (2009)
10. Engelmore, R.S.: Artificial Intelligence and Knowledge Based Systems: Origins, Methods and Opportunities for NDE. Review of Progress in Quantitative Nondestructive Evaluation 6 A, 1–20 (1987)

11. Fox, J., Ma, R.T.: Decision Support for Health Care : the PROforma Evidence Base. Informatics in Primary Care 14(1), 49–54 (2006)
12. Gardner, R.M., Pryor, T., Warner, H.R.: The HELP hospital information system: update 1998. International Journal of Medical Informatics 54(3), 169–182 (1999)
13. Han, P.K.J., Klein, W.M.P., Arora, N.K.: Varieties of uncertainty in health care: a conceptual taxonomy. Medical decision making : an international journal of the Society for Medical Decision Making 31(6), 828–38 (2011)
14. Hosobe, H., Satoh, K., Codognet, P.: Agent-based speculative constraint processing. IEICE Transactions on Information and Systems E90-D(9), 1354–1362 (2007)
15. Isern, D., Moreno, A.: Computer-based execution of clinical guidelines: A review. International Journal of Medical Informatics 77(12), 787–808 (2008)
16. Jamieson, S.: Likert scales: how to (ab)use them. Medical Education 38(12), 1217–1218 (2004)
17. Kalogeropoulos, D.A., Carson, E.R., Collinson, P.O.: Towards knowledge-based systems in clinical practice: Development of an integrated clinical information and knowledge management support system. Computer Methods and Programs in Biomedicine 72(1), 65–80 (2003)
18. Korb, K., Nicholson, A.: Bayesian Artificial Intelligence. CRC Press, London, 2 edn. (2003)
19. Latoszek-Berendsen, A., Tange, H., van den Herik, H.J., Hasman, a.: From clinical practice guidelines to computer-interpretable guidelines. A literature overview. Methods of information in medicine 49(6), 550–70 (2010)
20. Lipshitz, R., Strauss, O.: Coping with Uncertainty: A Naturalistic Decision-Making Analysis. Organizational Behavior and Human Decision Processes 69(2), 149–163 (1997)
21. McGuinness, D.L., Van Harmelen, F.: OWL Web Ontology Language Overview (2004), https://www.w3.org/TR/owl-features/
22. Miller, M., Kearney, N.: Guidelines for clinical practice: development, dissemination and implementation. International journal of nursing studies 41(7), 813–821 (2004)
23. Mishel, M.H.: The measurement of uncertainty in illness. Nursing research 30(5), 258–263 (1981)
24. Novais, P., Oliveira, T., Neves, J.: Moving towards a new paradigm of creation, dissemination, and application of computer-interpretable medical knowledge. Progress in Artificial Intelligence pp. 1–7 (2016)
25. Oliveira, T., Novais, P., Neves, J.: Representation of clinical practice Guideline components in OWL. In: Pérez, J.B., Hermoso, R., Moreno, M.N., Rodríguez, J.M.C., Hirsch, B., Mathieu, P., Campbell, A., Suarez-Figueroa, M.C., Ortega, A., Adam, E., Navarro, E. (eds.) Advances in Intelligent Systems and Computing, Advances in Intelligent Systems and Computing, vol. 221, pp. 77–85. Springer International Publishing (2013)
26. Oliveira, T., Novais, P., Neves, J.: Development and implementation of clinical guidelines: An artificial intelligence perspective. Artificial Intelligence Review pp. 999–1027 (2014)
27. Oliveira, T., Satoh, K., Novais, P., Neves, J., Hosobe, H.: A dynamic default revision mechanism for speculative computation. Autonomous Agents and Multi-Agent Systems pp. 1–40 (2016)
28. Open Clinical: Methods and tools for representing computerised clinical guidelines (2013), http://www.openclinical.org/gmmsummaries.html
29. Peleg, M., Boxwala, a.a., Bernstam, E., Tu, S., Greenes, R.a., Shortliffe, E.H.: Sharable representation of clinical guidelines in GLIF: relationship to the Arden Syntax. Journal of biomedical informatics 34(3), 170–181 (2001)
30. Peleg, M.: Computer-interpretable clinical guidelines: A methodological review. Journal of Biomedical Informatics 46(4), 744–763 (jun 2013)
31. Samwald, M., Fehre, K., de Bruin, J., Adlassnig, K.P.: The Arden Syntax standard for clinical decision support: Experiences and directions. Journal of biomedical informatics (feb 2012)

32. Seyfang, A., Miksch, S., Marcos, M.: Combining diagnosis and treatment using ASBRU. International journal of medical informatics 68(1-3), 49–57 (dec 2002)
33. Shahar, Y., Miksch, S., Johnson, P.: The Asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines. Artificial Intelligence in Medicine 14(1-2), 29–51 (1998)
34. Silberstein, S.: Clinical practice guidelines. Journal of neurosurgery. Pediatrics 25(10), 765–766 (2005)
35. Studer, R., Benjamins, V., Fensel, D.: Knowledge engineering: Principles and methods. Data & Knowledge Engineering 25(1-2), 161–197 (1998)
36. Terenziani, P., Montani, S., Bottrighi, A., Torchio, M., Molino, G., Correndo, G.: The GLARE approach to clinical guidelines: Main features. Studies in Health Technology and Informatics 101(3), 162–166 (jan 2004)
37. Toker, A., Shvarts, S., Perry, Z.H., Doron, Y., Reuveni, H.: Clinical guidelines, defensive medicine, and the physician between the two. American Journal of Otolaryngology - Head and Neck Medicine and Surgery 25(4), 245–250 (2004)
38. Tu, S.W., Campbell, J.R., Glasgow, J., Nyman, M.A., McClure, R., McClay, J., Parker, C., Hrabak, K.M., Berg, D., Weida, T., Mansfield, J.G., Musen, M.A., Abarbanel, R.M.: The SAGE Guideline Model: Achievements and Overview. Journal of the American Medical Informatics Association 14(5), 589–598 (2007)
39. Van der Heijden, M., Lucas, P.J.F.: Describing disease processes using a probabilistic logic of qualitative time. Artificial Intelligence in Medicine 59(3), 143–155 (2013)
40. Visscher, S., Lucas, P.J.F., Schurink, C.A.M., Bonten, M.J.M.: Modelling Treatment Effects in a Clinical Bayesian Network Using Boolean Threshold Functions. Artificial Intelligence in Medicine 46(3), 251–266 (2009)
41. Wang, D., Peleg, M., Tu, S.W., Boxwala, A.A., Ogunyemi, O., Zeng, Q., Greenes, R.A., Patel, V.L., Shortliffe, E.H.: Design and implementation of the GLIF3 guideline execution engine. Journal of Biomedical Informatics 37(5), 305–318 (2004)
42. Woolf, S.H., Grol, R., Hutchinson, A., Eccles, M., Grimshaw, J.: Potential benefits, limitations, and harms of clinical guidelines. BMJ : British Medical Journal 318(7182), 527–530 (feb 1999)
43. Young, O., Shahar, Y.: The Spock System : Developing a Runtime Application Engine for Hybrid-Asbru Guidelines. Artificial Intelligence Review 3581(1), 166–170 (2005)