Artificial Intelligence in Medicine 72 (2016) 22–41

Contents lists available at ScienceDirect

Artificial Intelligence in Medicine

journal homepage: www.elsevier.com/locate/aiim

META-GLARE: A meta-system for defining your own computer interpretable guideline system—Architecture and acquisition

Alessio Bottrighi*, Paolo Terenziani

Computer Science Institute, Dipartimento di Scienze e Innovazione Tecnologica, Universita' del Piemonte Orientale, Viale Teresa Michel 11, 15121 Alessandria, Italy

ARTICLE INFO

Article history:
Received 11 September 2015
Received in revised form 22 July 2016
Accepted 25 July 2016

Keywords:
Meta-modelling
Formalization and acquisition of health-care models
Computer interpretable guidelines

ABSTRACT

Context: Several different computer-assisted management systems of computer interpretable guidelines (CIGs) have been developed by the Artificial Intelligence in Medicine community. Each CIG system is characterized by a specific formalism to represent CIGs, and usually provides a *manager* to *acquire, consult* and *execute* them. Though there are several commonalities between most formalisms in the literature, each formalism has its own peculiarities.
Objective: The goal of our work is to provide a flexible support to the extension or definition of CIGs formalisms, and of their acquisition and execution engines. Instead of defining "yet another CIG formalism and its manager", we propose META-GLARE (META Guideline Acquisition, Representation, and Execution), a "meta"-system to define new CIG systems.
Method and materials: In this paper, META-GLARE, a meta-system to define new CIG systems, is presented. We try to capture the commonalities among current CIG approaches, by providing (i) a general manager for the acquisition, consultation and execution of hierarchical graphs (representing the control flow of actions in CIGs), parameterized over the types of nodes and of arcs constituting it, and (ii) a library of different elementary components of guidelines nodes (actions) and arcs, in which each type definition involves the specification of how objects of this type can be acquired, consulted and executed. We provide generality and flexibility, by allowing free aggregations of such elementary components to define new primitive node and arc types.
Results: We have drawn several experiments, in which we have used META-GLARE to build a CIG system (Experiment 1 in Section 8), or to extend it (Experiments 2 and 3). Such experiments show that META-GLARE provides a useful and easy-to-use support to such tasks. For instance, re-building the Guideline Acquisition, Representation, and Execution (GLARE) system using META-GLARE required less than one day (Experiment 1).
Conclusions: META-GLARE is a meta-system for CIGs supporting fast prototyping. Since META-GLARE provides acquisition and execution engines that are parametric over the specific CIG formalism, it supports easy update and construction of CIG systems.

1. Introduction

Clinical practice guidelines (CPGs) represent the current understanding of the best clinical practice. In recent years the importance and the use of CPGs are increasing in order to improve the quality and to reduce the costs of health care. ICT technology can further enhance the impact of CPGs. In the last twenty years, many different systems and projects have been developed to manage computer interpretable guidelines (CIGs), see, e.g., the collections [1–3]. A comparison among some existing systems is described in [4]. A "consensus" book of a part of the CIG community has been published in 2008 [3]. An excellent survey including many different approaches has recently been proposed by Peleg [5] (see Sections 2.1 and 2.2 for a brief analysis of related works).

Surveys and comparisons demonstrate that, though each approach proposes its own representation formalism, most approaches model CIGs as a task-network model (TNM): a hierarchical model of the guideline control flow as a network (graph) of specific tasks (represented by nodes). From the architecture point of view, most CIG systems provide specific support for at least two subtasks: (i) CIG acquisition and representation and (ii) CIG execu-

* Corresponding author.
E-mail addresses: alessio.bottrighi@uniupo.it (A. Bottrighi), paolo.terenziani@uniupo.it (P. Terenziani).

http://dx.doi.org/10.1016/j.artmed.2016.07.002
0933-3657/

tion. However, there are also important diverging features between the different CIG systems, also due to the fact that many of such systems are mostly research tools that evolve and expand to cover an increasing number of phenomena/tasks.

This is, for instance, the history of GLARE (Guideline Acquisition, Representation, and Execution), the prototypical system we have been building since 1996 in cooperation with ASU San Giovanni Battista in Turin, one of the major hospitals in Italy [6,7].

Unfortunately, at least in our experience, on the one hand it is quite frequent that due to the need to face a new real-world clinical guideline domain, some extensions to a CIG system are needed. On the other hand, such extensions are often quite complex and time consuming. In easy cases, the definition of the types of nodes used to represent actions have to be extended with new components (e.g., new attributes), so that the acquisition and consultation tools have to be extended to manage it (acquire it whenever a new instance of that type of node is created, store it, and show it during CIG consultation). In more complex cases, the new component may also affect node execution, so that the execution engine has to be modified, too (this was the case, for instance, when we added the treatment of "exceptions" to action nodes [8]).

In a complex CIG system (like GLARE) such extensions require a quite large amount of work, since different parts of the code system must be modified, and their interactions considered.

In the most complex cases, the amount of change in respect of a current system is so large that it might justify the development of a new system. For instance, this is the solution we were thinking of adopting when considering the use of GLARE for the education task. On the other hand, in research contexts it is very important to be able to easily and quickly extend systems, and to achieve fast prototyping when facing new domains and/or tasks.

With such goals in mind, we started to re-design GLARE. Initially, we started to analyse also many other CIG systems in the literature, with the goal of building a new *general* system, encompassing many of them. In particular, we were aiming to identify a *general formalism*, enclosing the "best features" of current ones. However, such an initial objective, though quite ambitious, has at least two main disadvantages:

(1) such a new general formalism would certainly be rich, general, and complex. These features somehow contrast with the fact that CIGs have to be acquired (with the cooperation of knowledge engineers) by expert-physicians, and have to be consulted/used by general practitioners. It is not reasonable practically to assume that physicians and general practitioners have to learn, understand (and possibly use, during CIG acquisition) very complex formalisms, possibly containing plenty of features that they do not need in the specific application domain they are considering;
(2) the long-term experience of artificial intelligence research has shown that there is no "perfect" formalism. Whatever general CIG formalism may be defined it can still require extensions, when facing new (unexpected by the formalism designers) phenomena.

As a consequence of (1) and (2), we decided to pursue a completely different and innovative goal: instead of defining "yet another new CIG formalism and system", we chose to devise a "meta-system", or, in other words, a shell supporting the definition of new CIG formalisms and systems (or facilitating the extensions of them). Our meta-system, called META-GLARE (META Guideline Acquisition, Representation, and Execution):

(i) makes "minimal" assumptions about the CIG formalisms (basically, it simply assumes that CIGs are represented through TNM)
(ii) provides general acquisition, consultation and execution engines, that are parametric over the specific CIG formalism being considered (in other words, the CIG formalism is an input of such engines)

The core idea of our meta-approach is:

(i) to define an open library of elementary components (e.g., textual attribute, Boolean condition, Score-based decision), each of which was equipped with methods for acquiring, storing, consulting and executing it;
(ii) to provide system-designers with an easy way of aggregating such components to define node and arc types (constituting the representation formalism of a new system);
(iii) to devise general and basic tools for the acquisition, consultation and execution of CIGs, represented by TNMs which are parametric over the node and arc types (in the sense that the definition of node and arc types are an input for such tools);
(iv) to couple each part of a system (e.g., elementary components, node definitions, system definition) with a declarative description of its main features and composition (automatically built during the system acquisition).

1.1. Advantages

In such a way, we achieve several advantages (further discussed in Section 8, through the treatment of some concrete examples of use of META-GLARE):

(1) the definition of a new system (based on a new representation formalism) is easy and quick. Using META-GLARE, a system designer can easily define her/his own new system, basically by defining its formalism: (i) the node types, (ii) the arc types (both are defined types as an aggregation of components from the library), and (possibly) the constraints on the way they can be connected in the graph. No other effort (e.g., building acquisition or execution modules) is needed;
(2) the extension of an existing system (through the modification of the representation formalism) is easy and quick. Using META-GLARE, a system designer can easily extend a system by defining and adding new node/arc types (by defining their components), or adding/deleting components to already existing types (with no programming effort at all; notice that the treatment of *versioning* is outside the goals of the current paper);
(3) user programming is needed only in case a new type of component (i.e., not already present in the library) is used (e.g., in the definition of a new node type). In such a case, the new type of component has to be added to the component library. However, the addition is modular and minimal: the system designer has just to focus on the added component, and to provide the code for acquiring, consulting, and (if needed) execute it. Notably, such programming is completely "local" and "modular": the new methods have to be programmed "in isolation" (without having to care about the rest of the system). In particular, neither the acquisition engine nor the execution engine have to be modified. Indeed, no modification to any software component in META-GLARE architecture (see Fig. 1 below) is required to integrate the new methods: META-GLARE automatically evokes them when needed during acquisition, consultation and execution.
(4) as a consequence, fast prototyping of the new (or extended) system is achieved.