Journal of Biomedical Informatics 59 (2016) 130–148

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

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

An architecture for a continuous, user-driven, and data-driven application of clinical guidelines and its evaluation

Erez Shalom [a,*], Yuval Shahar [a], Eitan Lunenfeld [b]

[a] Medical Informatics Research Center, Department of Information Systems Engineering, Ben Gurion University of the Negev, Beer-Sheva, Israel
[b] Soroka Medical Center, Department of OB/GYN, Faculty of Health Sciences, Ben Gurion University of the Negev, Beer-Sheva, Israel

ARTICLE INFO

Article history:
Received 1 February 2015
Revised 29 September 2015
Accepted 13 November 2015
Available online 23 November 2015

Keywords:
Clinical practice guidelines
Knowledge representation
Guideline execution engine
Telemedicine
Clinical decision support systems
Computer-assisted decision-making
Evaluation
Health care quality assurance
Guideline simulation

ABSTRACT

Objectives: Design, implement, and evaluate a new architecture for realistic continuous guideline (GL)-based decision support, based on a series of requirements that we have identified, such as support for continuous care, for multiple task types, and for data-driven and user-driven modes.

Methods: We designed and implemented a new continuous GL-based support architecture, PICARD, which accesses a temporal reasoning engine, and provides several different types of application interfaces. We present the new architecture in detail in the current paper. To evaluate the architecture, we first performed a technical evaluation of the PICARD architecture, using 19 simulated scenarios in the preeclampsia/toxemia domain. We then performed a functional evaluation with the help of two domain experts, by generating patient records that simulate 60 decision points from six clinical guideline-based scenarios, lasting from two days to four weeks. Finally, 36 clinicians made manual decisions in half of the scenarios, and had access to the automated GL-based support in the other half. The measures used in all three experiments were correctness and completeness of the decisions relative to the GL.

Results: Mean correctness and completeness in the technical evaluation were 1 ± 0.0 and 0.96 ± 0.03 respectively. The functional evaluation produced only several minor comments from the two experts, mostly regarding the output's style; otherwise the system's recommendations were validated. In the clinically oriented evaluation, the 36 clinicians applied manually approximately 41% of the GL's recommended actions. Completeness increased to approximately 93% when using PICARD. Manual correctness was approximately 94.5%, and remained similar when using PICARD; but while 68% of the manual decisions included correct but redundant actions, only 3% of the actions included in decisions made when using PICARD were redundant.

Conclusions: The PICARD architecture is technically feasible and is functionally valid, and addresses the realistic continuous GL-based application requirements that we have defined; in particular, the requirement for care over significant time frames. The use of the PICARD architecture in the domain we examined resulted in enhanced completeness and in reduction of redundancies, and is potentially beneficial for general GL-based management of chronic patients.

1. Introduction

1.1. Requirements for automated application of clinical guidelines

Clinical guidelines (GLs) are a powerful method for standardization and uniform improvement of the quality of the medical care [1]; however, free-text guidelines are often inaccessible at the point of care, and in any case, cannot be easily applied accurately to the patient at hand. Thus, there is a need for automated support for their specification and application at the point of care. The task of automated GL application was fairly well investigated in the recent years [2–5]. According to a study by Isern and Moreno [5], a computerized GL-based *Decision Support System* (DSS) infrastructure requires a central *Data Base* (DB), a central Medical *Knowledge Base* (KB) that stores the knowledge used during the task (sometimes modifying it) and a *DSS engine* that applies the knowledge to the data. Fig. 1 shows an abstract view of a typical knowledge-based DSS architecture provided by this approach: The medical knowledge of the GLs is stored and retrieved from a central KB library (examples of knowledge items stored in the KB are the

* Corresponding author at: Medical Informatics Research Center, Department of Information Systems Engineering, Ben-Gurion University of the Negev, P.O.B. 653, Beer-Sheva 8410501, Israel. Tel.: +972 54 4312565.

E-mail addresses: erezsh@bgu.ac.il (E. Shalom), yshahar@bgu.ac.il (Y. Shahar), lunenfld@bgu.ac.il (E. Lunenfeld).

http://dx.doi.org/10.1016/j.jbi.2015.11.006
1532-0464/

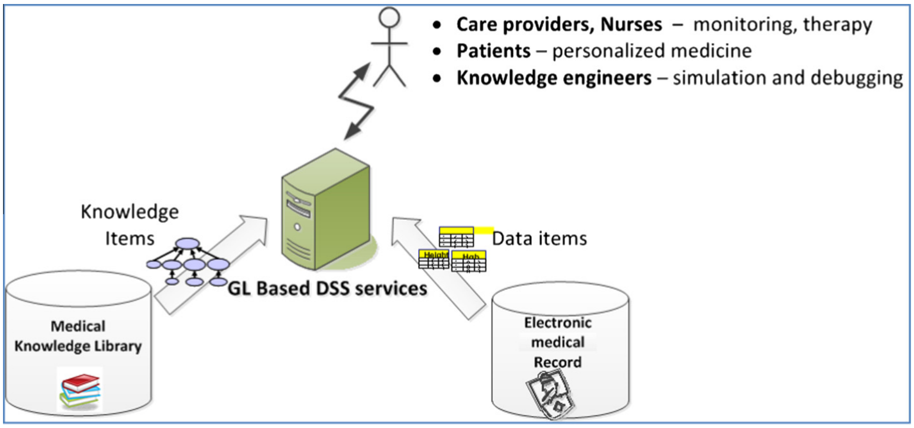

Fig. 1. An abstract view of the architecture of a knowledge-based medical decision-support system.

medical definition of "High BP" or definition of a plan to take a specific drug). The patient data are stored and retrieved from an *Electronic Medical Record* (EMR). The DSS engine applies the medical knowledge to the data to provide alerts and recommendations at the point of care to care providers (or nurses), to the patient (for example through messages sent to his mobile), or even to *Knowledge Engineers* (KEs), to debug or simulate the DSS engine.

However, building on our extensive experience with the DeGeL architecture [6], the Spock GL-application engine [7], the Uruz Web-based GL-specification tool [8,9] evaluations, and multiple prototypical GL applications, such as those described in Section 4, and on comments of medical domain experts, clinical users, and the current literature, we further formulated several additional key requirements for realistic automated GL application at the point of care, which none of the current frameworks fully support.

The main requirements that we have identified include:

(1) Provision of support for a *continuous* application of the GLs over significant stretches of time, providing recommendations when necessary.
(2) Verifying that the recommendations have *actually been carried out within the given time constraints*, based on evidence that exists in the EMR.
(3) Supporting a *data-driven, asynchronous application* (i.e., responding not only to entry of data during a session with the care provider, or to queries of the care provider, or to queries of the patient, but also to the arrival of data to the patient's record, from other sources and at other times).
(4) Provision of support through different application interfaces (APIs) for *different types of clinical actors* (e.g., nurses versus physicians versus patients), through a *scalable, distributed architecture*.
(5) Provision of *explanations*, regarding both the *procedural* (workflow-oriented) and *declarative* (data-interpretation oriented) aspects of the GL, which are accessible to the users, an important property for clinical DSSs [10].

1.2. Background: a comparison to common frameworks

Several existing frameworks provide various types of solutions to the problem of specification and application of clinical GLs; examples include EON [11,12], GLIF3 [13,14], GASTON [15,16], ProForma [17,18], GLARE [19,20], NewGuide [21,22], SAGE [23], PRODIGY [24], Asbru Interpreter [25], SPOCK [7], and Health Care Services (HeCaSe2) [26,27]. Table 1 categorizes the properties of several of these leading research frameworks for GL application, with respect to the requirements introduced in the previous section.

Note that most of the listed frameworks have only partially demonstrated full-fledged support for continuous GL application over time. This includes cases in which the framework's underlying language supports, in theory, a specification of continuous GL application over time, but we have not found any detailed implementation or demonstration of a complex GL applied over time using the framework. In addition, most of the frameworks do not attach to each recommendation an effective explanation that can justify to the user why a particular DSS recommendation was suggested. Several frameworks do not include, or include only partially, an API that support multiple tasks, such as debugging or simulation, in addition to GL application. Finally, note that most frameworks provide only partial support for the verification of the actual application of an accepted recommendation, e.g., by enabling the user to explicitly accept the recommendation. But they do not actually verify that the expected results of applying

Table 1
Comparison of the guideline-based decision-support frameworks; see Section 1.1 for the description of each requirement; √x means partially supported.

Framework name	Req #1	Req #2	Req #3	Req #4	Req #5	Functional evaluation of the GL application process	Evaluation of effect on clinical decision making
EON	√x	√x	X	√x	X	✓	✓
GLIF3	√x	√x	✓	✓	√x	✓	X
GASTON	√x	√x	✓	✓	X	✓	X
Proforma	✓	√x	✓	✓	✓	✓	✓
GLARE	✓	√x	✓	✓	X	✓	X
NewGuide	√x	s	✓	√x	X	✓	√x
SAGE	√x	√x	✓	✓	X	✓	X
PRODIGY	√x	√x	X	✓	X	✓	X
Spock	√x	X	X	√x	X	✓	X
Asbru Interpreter	√x	X	X	X	X	✓	X
Health Care Services (HeCaSe2)	√x	√x	✓	X	X	✓	X