Webifying the Computerized Execution of Clinical Practice Guidelines

Tiago Oliveira[1], Pedro Leão[2,3,4], Paulo Novais[1] and José Neves[1]

[1]CCTC/Department of Informatics, University of Minho, Braga, Portugal
[2] School of Health Sciences, University of Minho, Braga, Portugal
[3]Life and Health Sciences Research Institute, Hospital of Braga, Braga, Portugal
[4] ICVS/3B's - PT Government Associate Laboratory, Braga/Guimarães, Portugal
[1]{toliveira, pjon, jneves}@di.uminho.pt
[2]pedroleao@ecsaude.uminho.pt

Abstract. The means through which Clinical Practice Guidelines are disseminated and become accessible are a crucial factor in their later adoption by health care professionals. Making these guidelines available in Clinical Decision Support Systems renders their application more personal and thus acceptable at the moment of care. Web technologies may play an important role in increasing the reach and dissemination of guidelines, but this promise remains largely unfulfilled. There is a need for a guideline computer model that can accommodate a wide variety of medical knowledge along with a platform for its execution that can be easily used in mobile devices. This work presents the CompGuide framework, a web-based and service-oriented platform for the execution of Computer-Interpretable Guidelines. Its architecture comprises different modules whose interaction enables the interpretation of clinical tasks and the verification of clinical constraints and temporal restrictions of guidelines represented in OWL. It allows remote guideline execution with data centralization, more suitable for a work environment where physicians are mobile and not bound to a machine. The solution presented in this paper encompasses a computer-interpretable guideline model, a web-based framework for guideline execution and an Application Programming Interface for the development of other guideline execution systems.

Keywords: Computer-Interpretable Guidelines, Clinical Decision Support, Framework, Web.

1 Introduction

Clinical Practice Guidelines (CPGs) are systematically developed statements that contain recommendations to assist health care professionals and patients in specific clinical circumstances [1]. Their main goals are to provide patient specific advice, to reduce variations in medical practice and to promote cost containment, through efforts towards the improvement of efficiency and quality. From the appearance of CPGs as information vehicles of medical consensus groups and evidence-based medicine in the late 1970s until the present day their use has become widespread, being regarded by most as useful tools in health care delivery. However, there are some issues that are continuously raised by the medical community [2]. Many believe that paper-based guidelines are difficult to consult at the moment of care. Others think they are an instrument for cookbook medicine and are not convinced that their use leads to better care. Fortunately, the advent of Clinical Decision Support Systems (CDSSs) as tools for information management and providing patient specific recommendations offered a medium through which guideline appliance can become more acceptable, patient tailored and interactive [3]. These are the features that make CPGs implementable during care delivery. As will be seen far ahead, Computer-Interpretable Guideline (CIG) implementations appeared in the early 1990s and have proliferated since then.

In the early 2000s the emergence of t Web 2.0 technologies created yet another opportunity for the evolution of CIGs. It was a change in how the web is perceived by both users and developers, accompanied by the adoption of interaction and participation as fundamental aspects of online activity. The concept of the web as an integrating platform with rich internet applications offers software above the level of a single device. For CIGs this translates into the possibility of their being available anywhere, anytime. This work is an introduction to the *CompGuide* web-based framework for CIG execution.

The article is organized as follows. Section 2 provides insights on current CIG development and the relevant work done in the field. Section 3 gives a brief explanation of the model used for CIG representation. The framework is presented in Section 4. Finally, Section 5 provides some conclusions about the work done so far and future considerations for the work ahead.

2 Relevant Work

Since the 1990s many researchers have developed and proposed CIG models to represent guidelines in a computer-interpretable language. Arguably, the most relevant are Arden Syntax [4], the Guideline Interchange Format (GLIF) [5], Asbru [6], PROforma [7] and the Standards-based Active Guideline Environment (SAGE) [8]. Apart from Arden Syntax, which is a model specifically for encoding small fragments of clinical knowledge in the form of rules, the majority of models employ some kind of task-oriented network to represent and display guideline knowledge. The Task Network Model (TNM) appears to be the one that best fits the information conveyed by guidelines, allowing for a clear separation between procedural knowledge, i.e. the relative order of recommendations, and medical knowledge. A model is usually accompanied with an execution engine which is responsible for interpreting the clinical constraints and temporal constraints placed on tasks. Tools such as Arezzo^{TM} (for PROforma), the Digital Electronic Guideline Library (DeGeL) (for Asbru), the Guideline Execution Engine (GLEE) (for GLIF3) and SAGEDesktop are used to provide recommendations interactively and store execution traces of guidelines represented in their respective models. Normally, the execution engines comprise a desktop client application that remotely connects to a server which provides a guideline repository. This tendency for the development of desktop applications is evident in Isern and Moreno's review work [9] on computer-based execution of CPGs. Despite the virtues of this type of structure and desktop applications, one is forced to consider that CIG development is not taking advantage of the web's full capabilities to increase the reach and availability of CPGs, their level of dissemination, and their scrutiny. A comparative analysis of current approaches to CIGs was done in previous work [10], where a comparative table regarding their main features is provided.

In a recent review article [11], Peleg indicated the development of ubiquitous CIG-based systems as an emergent trend. Through web-based and smartphone/tablet interfaces, these systems should be able to provide ever accessible clinical decision support. There are projects that have already started to develop their CIG solutions under the concept of guidelines as services. The work presented herein assimilates these ideas, and implements a Service Oriented Architecture (SOA) to make available the functionalities that enable the execution of CIGs. It comprises a guideline representation model, a server application that provides a set of guideline-related services and a web interface that makes full use of these services.

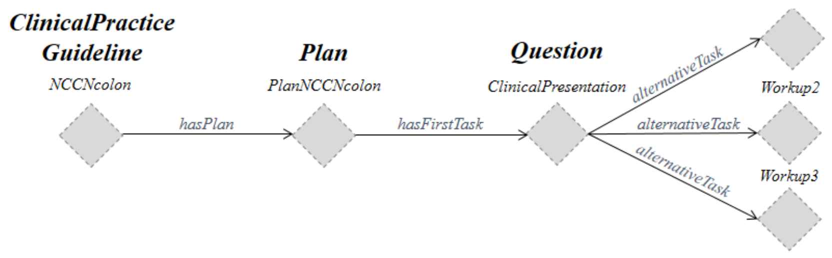


Fig. 1. Initial formalization of the NCCN Guideline for Colon Cancer in CompGuide ontology. A guideline is represented as a network of individuals connected by object properties.

3 Computer-Interpretable Guideline Model

The *CompGuide* ontology model for CIGs, which is used in this work, is expressed in OWL-Description Logic (OWL-DL) version 2 [13], and it has been previously presented in [12]. Protégé was used as an ontology development tool and in the creation of guidelines. In the ontology, a CPG is represented as an individual of the class *ClinicalPracticeGuideline*, which has a set of data properties to express administrative information and object properties to connect it to individuals of other classes. This set up may be seen in Fig. 1 which is the initial formalization of the National Comprehensive Cancer Network (NCCN) Guideline for Colon Cancer according to the ontology. The TNM is implemented in the form of four classes of tasks: *Plan***,** a collection of tasks containing any number of other tasks, including other plans; *Action***,** a task performed by a health care agent, namely a clinical procedure, a clinical exam, a medication recommendation or a non-medication recommendation; *Question*, an enquiry task to obtain information about the patient; and *Decision*, a reasoning task about the state of a patient which implies the choice between two or more options, yielding a conclusion which is then used to update the state of the patient.

Similarly to what is expressed in Fig. 1, different tasks are represented by individuals of the above-mentioned classes. Being a workflow representation, there have to be control structures to define the relative order between tasks. A guideline has a main *Plan* which contains all the tasks. The individual corresponding to this *Plan* has, in turn, an object property that points to its first task. Then, the previous tasks always indicate those which follow. It is possible to define sequential tasks, tasks which should be executed at the same time (parallel tasks) and alternatives in the guideline workflow (alternative tasks). For parallel tasks it is also possible to define synchronization tasks where the workflow reunifies after splitting.

A CIG model requires the appropriate constructs for the definition of clinical constraints. In this regard, *CompGuide* offers the possibility to define *TriggerConditions* which are used to specify the terms regarding the patient state that dictate the choice of an alternative task. Other clinical constraint classes are *PreCondition*, which specifies the conditions that must be met before executing a task, and *Outcome*, the expected result of a task in terms of the alterations it produced in the patient state. It is also possible to define, in *Decisions*, the options to choose from and rules associated with them.

Temporal restrictions are also an important element of medical algorithms. Thus, *CompGuide* provides *Periodicity* and *Duration* classes. The former may be used to express from when to when a task should be executed and/or its number of repetitions. Through *Periodicity* it is also possible to define stop conditions for a cyclic task and, in the event of these stop conditions holding true, the task the guideline execution should move to, which is a stop condition task. The *Duration* indicates how long a task should last.

4 CompGuide Framework

The CompGuide framework was created in order to provide tools capable of automating the functionalities offered by the expressiveness of the ontology, to automatically process workflow control structures, clinical constraints and temporal restrictions. The following briefly introduces the tools and technologies used in the development, explains the framework's architecture and addresses the most significant aspects regarding guideline execution.

4.1 Tools and Technologies

The tools and technologies which were used were chosen by taking into consideration the requirements of the application's architecture, namely its strong web component, as mentioned in Section 2. Accordingly, web services were used as the preferential form of communication. Their usage offers expandability and the possibility to improve services without compromising others. For their lightness and ease of access, Representational State Transfer (REST) Web Services were the chosen service model. To stay aligned with the overall goal, the framework was developed as a Web Application

Programming Interface (Web API) to enhance multi-platform implementation capabilities.

The access to guidelines in the OWL ontology file is done through the Java OWL-API, developed and maintained at the University of Manchester. It provides an easy way to create, manipulate and serialize OWL ontologies. The web application that serves as user interface was developed using Java Server Pages (JSP). The data exchanged with the server is in JavaScript Object Notation (JSON), a text-based open standard for data interchange. It offers simple parsing and, at the same time, is compact when transferring small amounts of data.

The user, patient and execution information is stored in a MySQL relational database. The applications are deployed in a JBoss application server.

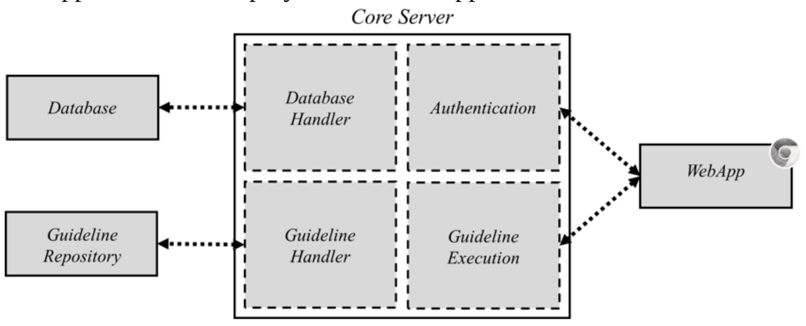


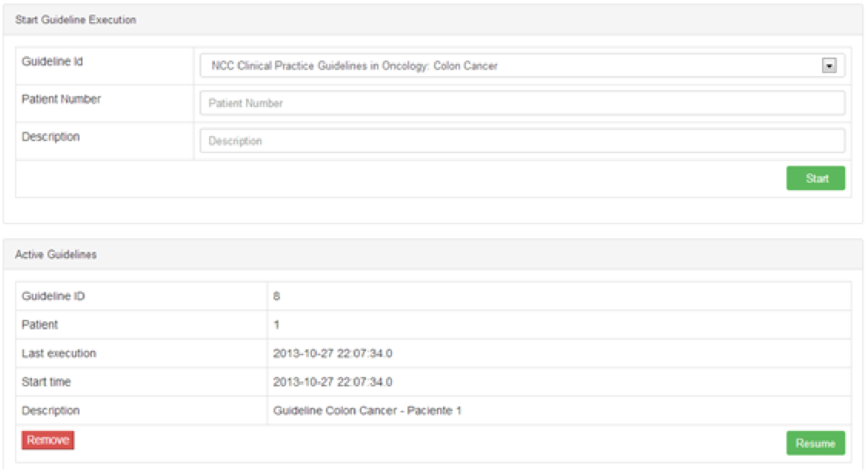
Fig. 2. Architecture of the *CompGuide* framework with the Core Server connecting the data storage components to the web application.

4.2 Architecture

The framework's architecture is depicted in Fig. 2. It is composed of four separate main components. The communication between the main applications, in particular with the web application, is enabled by web services. The data storage components feature a *Database* model to store user, patient and execution information, and the *Guideline Repository* which is an OWL file in order to store CPGs. The *Core Server* uses JBoss to provide the services required for guideline execution to external applications. It is responsible for all the database manipulation and guideline execution control. It takes advantage of JBoss's features by implementing session controllers, user authentication and publication of web services (accessible through HTTP methods). Within the *Core Server* there are different components which work under the web service layer, being seamless for the user interface applications. Said components comprise the following modules:

- *Authentication*: the module provides both authentication and authorization. Two types of users are defined: *admin* and *user*. While a *user* has full access to manipulate user and patient information (add, remove and edit) and execute guidelines, an *admin* has access only to the guideline execution functionality. The authentication process is done through a specific web service which sends a token as response. This token is then used to request the other services until expiring and ensures that the level of access of the user is respected. The token verification is handled by a filter.

- *Guideline Handler*: this module does the retrieval of guidelines from the *Guideline Repository*. It includes a set of functions to fetch the information for each class on the guideline ontology. In big hand coded guidelines there are often mistakes, such as missing components and properties, hence, a syntactic verification tool was created so as to guarantee the validity of a guideline in this regard.
- *Guideline Execution*: guideline execution is done entirely in the *Core Server* by this module. If offers two distinct services, both responsible for computing the next task in the guideline algorithm as well as providing the information featured in it. The difference between the two is that one of them allows to skip a task if the physician desires to do so, while the other does not. When the task is executed, the module stores this event in order to restore the state of the execution to previous moments. This enables the resuming of a suspended execution trace for a patient.
- *Database Handler*: it contains all the updates, inserts and selects required by the *Core Server* to handle stored data.


Start Guideline Execution

Guideline Id	NCC Clinical Practice Guidelines in Oncology: Colon Cancer
Patient Number	Patient Number
Description	Description

Active Guidelines

Guideline ID	8
Patient	1
Last execution	2013-10-27 22:07:34.0
Start time	2013-10-27 22:07:34.0
Description	Guideline Colon Cancer - Paciente 1

Fig. 3. Interface of the guideline management suite showing the active guidelines with the option to resume their execution and the new guideline ready to be deployed.

The implemented *Database* model stores all the user and patient information combined with the information produced by the interface application. This allows for the mapping of user and patient records to every guideline execution instance. The structure is based on an OpenEHR archetype model [13]. The information is saved as *observations* (relevant states of the patient retrieved through *Question* tasks) and *actions* (exams, medication, non-medication recommendations and procedures).

The *Web App* aims at collecting information from and delivering information to the user. It provides a *control panel* which enables the editing of the personal information of users and patients, and a *management suite* (as shown in Fig. 3) that offers the possibility to start and resume the execution of guidelines. When starting a new guideline, the user must specify the patient to whom the guideline will be applied. The interface for the guideline execution covers all the tasks mentioned in Section 3, with tasks being presented as they are computed by the execution engine.

4.3 Aspects of Guideline Execution

In CIG execution, there are essentially three types of verification that execution engines must perform. They are task ordering verifications, clinical constraint verifications and temporal restriction verifications.

The model allows the representation of different modalities of task performing, from sequential tasks and parallel tasks to alternative tasks. Mixing these types together, which may occur in real guideline algorithms, significantly increases the complexity of the programming necessary to handle these situations. As an example, a parallel task may be followed by another set of parallel tasks or alternative tasks which, in turn, can also be followed by other such tasks. This raises issues in keeping their synchronization points in check. To tackle them, a *task controller* was developed for storing information at three levels: the task, the plan, the parallel tasks and the alternative tasks. It ensures that the execution engine follows a plan and that all parallel and alternative tasks are synchronized. The patient state is built through *Question* tasks. The information is collected and stored as an *observation*. Whenever it is necessary to check trigger conditions for alternative tasks or validate rules in *Decision* tasks, the values of the clinical parameters contained in them are searched in *observations*. A *PreCondition* is verified before proposing a task and an *Outcome* after performing it. *StopConditions* are verified at every iteration of a periodic task. Every task entry in the *Database* has a timestamp. This is used to store the moment when a specific task is performed and to control the different temporal restrictions. When a temporal restriction is met, a warning is issued to the user.

5 Conclusions and Future Work

Leaving the heavier processing, such as task computation, to the server proves to be an effective implementation of CIG execution, removing the need to exchange big chunks of information. Comparing with existing systems, the hereby presented architecture offers some advantages, in particular it provides an API to access an OWL CPG ontology that other developers can use in their own application, it is easy to access its functionalities given the service-oriented architecture, and it offers the possibility to easily develop other user-interfaces. It allows remote guideline execution with data centralization, more suitable for a work environment where physicians are mobile and not bound to a machine. Nevertheless the usefulness of such a framework, its development is still at an early stage. The *Core Server* requires more modules that implement functionalities that are missing such as guideline creation and terminology services. The integration of Unified Medical Language System (UMLS) features is within the scope of the project, as is the mapping of the information produced by guideline execution to standards of medical information exchange, such as the Health Level Seven (HL7)

Clinical Document Architecture. The original contributions of this work are: a computer-interpretable guideline model which can be used for any type of guideline and does not require knowledge on any programming languages; a web-based platform for the execution of clinical practice guidelines in a simple and intuitive way; and an API that developers can use in their own implementation

In the long run, the information stored by the system may be useful in order to assess how guidelines cope with situations and if physicians conform to them.

Acknowledgements

This work is part-funded by ERDF - European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project FCOMP-01-0124-FEDER-028980 (PTDC/EEI-SII/1386/2012). The work of Tiago Oliveira is supported by doctoral grant by FCT (SFRH/BD/85291/2012).

References

[1] M. J. Field and K. Lohr, *Guidelines for Clinical Practice: From Development to Use*. Washington DC: The National Academy Press, 1992.

[2] S. H. Woolf, R. Grol, A. Hutchinson, M. Eccles, and J. Grimshaw, “Potential benefits, limitations, and harms of clinical guidelines,” *BMJ Br. Med. J.*, vol. 318, no. 7182, pp. 527–530, 1999.

[3] A. Latoszek-Berendsen, H. Tange, H. J. van den Herik, and a Hasman, “From clinical practice guidelines to computer-interpretable guidelines. A literature overview.,” *Methods Inf. Med.*, vol. 49, no. 6, pp. 550–70, Jan. 2010.

[4] M. Samwald, K. Fehre, J. de Bruin, and K.-P. Adlassnig, “The Arden Syntax standard for clinical decision support: Experiences and directions.,” *J. Biomed. Inform.*, Feb. 2012.

[5] M. Peleg, A. A. Boxwala, O. Ogunyemi, Q. Zeng, S. Tu, R. Lacson, E. Bernstam, N. Ash, P. Mork, L. Ohno-Machado, and others, “GLIF3: the evolution of a guideline representation format.,” in *Proceedings of the AMIA Symposium*, 2000, p. 645.

[6] M. Balser, C. Duelli, and W. Reif, “Formal Semantics of Asbru – An Overview,” *Science (80-.).*, 2002.

[7] E. Vier, J. Fox, N. Johns, C. Lyons, A. Rahmanzadeh, and P. Wilson, “PROforma : systems,” *Comput. Methods Programs Biomed.*, vol. 2607, no. 97, 1997.

[8] P. Ram, D. Berg, S. Tu, G. Mansfield, Q. Ye, R. Abarbanel, and N. Beard, “Executing clinical practice guidelines using the SAGE execution engine.,” *Stud. Health Technol. Inform.*, vol. 107, no. Pt 1, pp. 251–5, Jan. 2004.

[9] D. Isern and A. Moreno, “Computer-based execution of clinical guidelines: a review,” *Int. J. Med. Inform.*, vol. 77, no. 12, pp. 787–808, 2008.

[10] T. Oliveira, P. Novais, and J. Neves, “Development and implementation of clinical guidelines: An artificial intelligence perspective,” *Artif. Intell. Rev.*, Mar. 2013.

[11] M. Peleg, “Computer-interpretable clinical guidelines: A methodological review.,” *J. Biomed. Inform.*, vol. 46, no. 4, pp. 744–63, Aug. 2013.

[12] T. Oliveira, P. Novais, and J. Neves, “Representation of Clinical Practice Guideline Components in OWL,” in *Trends in Practical Applications of Agents and Multiagent Systems SE - 10*, vol. 221, J. B. Pérez, R. Hermoso, M. N. Moreno, J. M. C. Rodríguez, B. Hirsch, P. Mathieu, A. Campbell, M. C. Suarez-Figueroa, A. Ortega, E. Adam, and E. Navarro, Eds. Springer International Publishing, 2013, pp. 77–85.

[13] P. Gutiérrez, “OpenEHR-Gen Framework,” 2010.