A Unified System for Clinical Guideline Management and Execution

António Silva[1(✉)], Tiago Oliveira[2(✉)], Filipe Gonçalves[1(✉)], José Neves[1(✉)], Ken Satoh[2(✉)], and Paulo Novais[1(✉)]

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

Abstract. There are several approaches to Computer-Interpretable Guidelines (CIG) representation and execution that offer the possibility of acquiring, executing and editing CPGs. Many CIG approaches aim to represent Clinical Practice Guidelines (CPGs) by computationally formalising the knowledge that they enclose, in order to be suitable for the integration in Clinical Decision Support Systems (CDSS). However, the current approaches for this purpose lack in providing a unified workflow for management and execution of CIGs. Besides characterising these limitations and identifying improvements to include in future tools, this work describes the unified architecture for CIG management and execution, a unified approach that allows the CIG acquisition, editing and execution.

Keywords: Artificial intelligence in medicine
Computer-Interpretable Guidelines
Clinical Decision Support Systems · CIG execution engines · CIG tools
CIG representation and execution

1 Introduction

Currently, in the health sector, there is a growing need for systems that provide features that allow the representation and execution of Clinical Practice Guidelines (CPGs) computationally in order to be able to provide decision support through patient-specific recommendations based on CPGs for automatic interpretation, called Computer-Interpretable Guidelines (CIGs). CIG representation languages serve two different purposes: modelling of the medical knowledge encapsulated in the paper-based CPGs and capturing the workflow structure of the paper-based protocol and making the necessary mappings between the workflow variables and their counterpart variables in the domain knowledge. However, there are some obstacles to the deployment of CPGs in Clinical Decision Support Systems (CDSSs) for daily use, making it difficult to represent these documents computationally. Most of the current CPGs were not designed to be digitally

Á. Rocha et al. (Eds.): WorldCIST'18 2018, AISC 746, pp. 810–819, 2018.
https://doi.org/10.1007/978-3-319-77712-2_76

represented and computer-interpretable since they imply complex and intricate instructions, complex execution structures and the manipulation of too many variables leading to non-deterministic and complex algorithms [12].

Although current solutions provide ways to represent CPGs computationally, by offering tools to create and execute them, they lack in providing a system that allows a unified workflow from creation to the execution of guidelines. The present work discloses a unified tool, which allows representing CPGs as a CIG through the CompGuide Editor and provides a system that promotes a better integration of CPG recommendations in the daily life of health care professionals by producing an agenda containing clinical recommendations, allowing the execution of CIGs [6,9]. The work described herein presents the CompGuide ontology [8], a model which provides a representation of CPGs and the CompGuide Editor, a plugin that performs the role of managing the creation and editing of CIGs. By connecting these two components, we intend to propose a unified architecture for CIG acquisition, editing and execution. Thus, the contributions featured in this work are: the full integration of the stages in a CIG development life cycle, direct expedience of CIGs to a clinical situation and a unified workflow for the acquisition and customisation of CIGs.

The present work is organised as follows. Section 2 describes related work regarding systems for CIG execution and tools for CIG creation and editing. In Sect. 3, we present a CompGuide architecture proposal for CIG management and execution as well as the contributions for the deployment of CPGs in CDSSs. Section 4 describes the functionalities supporting care with examples of CIG management and execution. Finally, Sect. 5 presents conclusions about the work developed so far and future work considerations.

2 Existing Systems for CIG Management and Execution

There are two purposes for CIG representation languages: support the acquisition and editing of CPGs, either manually or semi-automatically, and the modelling of the workflow structure of CPGs for CIG execution.

2.1 Existing CIG Tools

Some of the most important modelling editors and CIG managers are Protégé Desktop [7], SAGE Workbench [1], and Asbru View [14].

Protégé [5] is an open source ontology development and knowledge acquisition environment. It is a graphical Java tool, which provides an extensible architecture for the creation of customised knowledge-based tools and assists users in the construction of large electronic knowledge bases. knowledge-based tools and applications. Protégé is used to author guidelines in various models. Part of the modelling can be accomplished using predefined graphical symbols. These symbols are arranged in a diagram and linked by graphs. The underlying data is entered by forms [4]. Protégé ontologies can be exported into a variety of formats including RDF(S), OWL, and XML Schema [10].

The SAGE Workbench [1] is a complete, self-contained environment that uses the SAGE guideline model. This model encodes guideline knowledge needed to provide situation-specific decision support and use standardised components for interoperability. SAGE Workbench includes a suite of tools to create, view, edit, and validate SAGE guidelines that conform to the format of the *SAGE* model and that are executable by the SAGE Execution Engine. One of its key approaches is to integrate guideline-based decision support with the workflow of the care process. In addition, *SAGE* is recognised as one of the improved clinical decision support architectures with its large coverage. *SAGE* includes a knowledge authoring tool based on *Protégé*.

Asbru View is a tool to make Asbru accessible to physicians and to give any user an overview of a plan hierarchy. Asbru View is based on visual metaphors to make the underlying concepts easier to grasp. This was done because not only is the notation foreign to physicians, but also the underlying concepts. In other words, Asbru View is a graphical user interface for viewing, creating and modifying Asbru plans. It is based on different views of different aspects of the plans [14]. These views are Topological View (TopoView) and Temporal View (TempView). The Topological View mainly displays the relationships between plans, without a precise timescale. The basic metaphor in this view is the running track. The Temporal View concentrates on the temporal dimension of plans and conditions. This view allows seeing the details of the temporal extensions of plans.

Most tools present a form of graphical view of guidelines. Although the analysed tools are well organized, some of them lack simplicity. In the cases of Protégé Desktop and SAGE Workbench, despite having many useful features available, the massive number of menus they display make the user lose a lot of time in understanding the different functionalities. So, the simplicity of the interface should be prioritised. As for the existence of a Drag-and-Drop feature, it is an important element these days, enabling the management of a CIG visual layout and more comprehensive user form. However, the tools studied lack in providing such functionality. Additionally, one of the features that new platforms should present is the capacity to deploy automation data since it is absent from existing tools. With the capacity to deploy automation data, the user only implements the data relevant to the instance, leaving the less important details to the responsibility of the system. Another important feature is the ability to import or export CIGs stored locally or in a cloud. Once the information held in clouds, it will allow users to access all this data anywhere, anytime.

2.2 Existing Systems for CIG Execution

Several guideline description languages exist to create computer-assisted tools for executing CPGs that are aimed at representing clinical knowledge. A model is usually accompanied by an execution mechanism that is responsible for interpreting the clinical constraints and temporal constraints placed on tasks.

The Spock Engine [16] was developed to enable the execution of CIGs in the Asbru model [11]. The Spock system was integrated into the Digital electronic

Guideline Library (DeGeL) framework and can execute guidelines represented in Hybrid-Asbru intermediate representation. It incorporates an inference engine that can retrieve data from the patient's electronic medical record. It is designed as a modular client-server architecture, which facilitates the gradual conversion of clinical guidelines from text to a formal representation in a selected target guideline ontology. It consists of classes and interfaces encapsulating the semantics of how to apply a hybrid Asbru guideline and a synchronizer that establishes the communication with external systems. This execution engine maintains a runtime application state, storing a guideline application log that contains several data structures for each application-instance. They include a queue of scheduled awaiting tasks, and the list of recommendations already issued.

GLEE [15] is an execution engine based on the third version the Guideline Interchange Format (GLIF3), which allows a better integration with clinical information systems. This approach was developed to reflect a flowchart of structured and scheduled steps that represent clinical decisions and actions. It was designed using the Model-View-Control (MVC), which means that it is divided into three layers namely *data*, *business logic* and *user interface*. The *data* layer contains an electronic medical record with patient data, a guideline repository, and a clinical event monitor that allows the execution of CIGs driven by clinical events. This layer is responsible for providing all necessary data with the upper layers. The *business logic* layer defines a guideline implementation system; it means an execution model of guidelines that support the functions defined in the GLIF3 model. This layer is responsible for managing and control the access to EMR data. All the information is displayed in the *user interface* layer which is responsible for supporting the interactions between users and the application. The execution engine records every clinical parameter from a patient during the execution of a CIG, suggesting actions to be performed.

The GLARE Execution Engine is based on the Guideline Acquisition, Representation and Execution (GLARE) model [13]. It defines a graph-based framework that displays CPGs, where a clinical action is represented by a node. It is possible to define atomic actions that represent simple tasks, such as *queries* to obtain external information, work actions that represent medical procedures, decision actions as a set of conditions to select alternatives and conclusions that describe the decision making. Similarly to the other systems, GLARE also defines three abstraction layers. In this case they are called *system*, *eXtensible Markup Language* (*xml*), and *dbms*. The System level includes the acquisition and execution modules. The XML level is responsible for exchanging data between the System level and the DBMS level. DBMS is the lowest level, responsible for establishing a physical connection between higher levels and databases where information for the creation and execution of clinical protocols is stored. This information includes open instances of clinical protocols, a repository of protocols and medical records of the patients.

All these systems use structures and well-defined languages that provide mechanisms to represent and analyse CPGs. Moreover, they are also well-structured models providing features for accessing the data through an electronic

medical record, modelling the business logic through runtime engines that represent the CIG knowledge and provide guideline repositories. They also support modules that describe the context, as in the case of the GLARE Execution Engine, mainly concerning the availability of the resources, at the health care institution where CIG deployment is taking place. Their objective is to generate case-specific recommendations based on CIG instructions and patient data items. In these systems, the role of the execution engine is crucial since these components are responsible for following the constraints of the clinical workflow and the temporal constraints placed on tasks, producing clinical recommendations based on task enactment times. Some above-mentioned applications for CIG execution exist in the form of web-based client-server applications, with the intelligence engine placed on the server side. However, others exist in the form of desktop applications, which is an obstacle for integrating CIG recommendations in the daily routine of health care institutions.

3 An Architeture for CIG Management and Execution

The present work provides a system not only for the execution of CIGs but also their acquisition and editing, in a unified workflow. CompGuide provides a CIG execution server [9] and a CompGuide plugin editor [3], aiming at reusing and integrating knowledge components. The unified architecture is shown in Fig. 1.

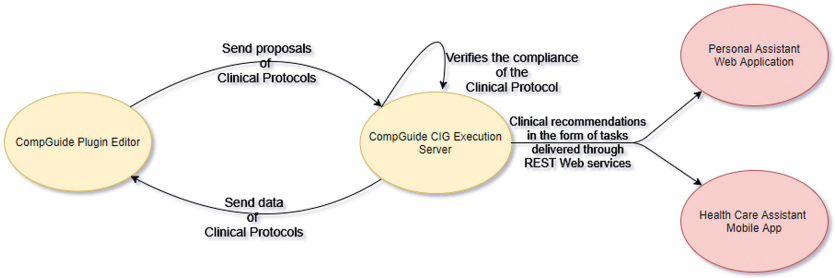

Fig. 1. Architecture for the management and execution of CPGs.

The CompGuide editor is a *Protégé Desktop* plug-in interface that allows the creation and editing of CIGs in a quick and simple way. The idea of creating a *Protégé Desktop* plug-in emerged from the need to create software capable of implementing all the features offered by the *Protégé Desktop* application (specifically the functionality of managing the data of an ontology through the use of a graphical interface) along with the creation of new features.

The tool offers three views: OntoGraf View that allows the graphical representation of the ontology; the Individuals by Type View, which shows all the individuals (sorted by respective OWL classes) saved in the ontology; and CompGuide Wizard Options View, which has the set of features to manage the

individuals and their relations, plus download/upload the CompGuide ontology file. This view also shows the total number of individuals saved in the loaded ontology. So, it is intended to reuse this plugin to allow the management of clinical protocols, providing access to features, such as creation, editing and deletion of data concerning these documents.

After the health professional finishes the editing or creation of the ontology, the proposal will be sent to the *CIG Execution Server*. The *CIG Execution Server* contains a *Guideline Repository* [9], so this component will be responsible for verifying the compliance of the proposal and saving guideline files in the *Guideline Repository*. Furthermore, whenever necessary, the *CIG Execution Server* must provide CPG data to the CompGuide plugin editor, in order to ensure that the health professional works on the last version of the guideline.

The clinical protocols will be available to apply for a patient, through the *Personal Assistant Web Application* or *Health Care Assistant Mobile App*. Figure 2 shows how the health professional can apply a CPG to a patient through the *Personal Assistant Web Application*.

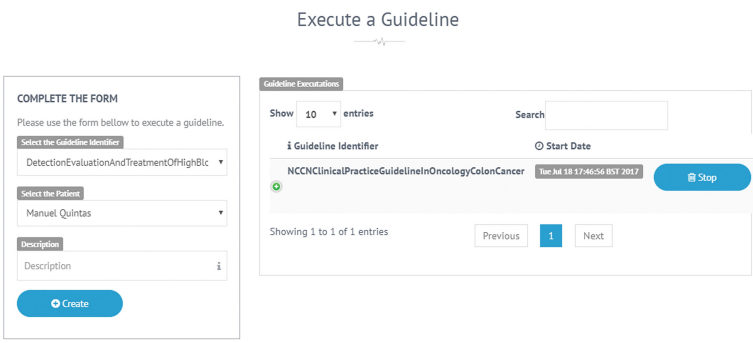

Fig. 2. Web page with clinical protocols to apply for a patient.

These two components allow the health professionals to consult and monitor the progress of patients as well as the clinical recommendations wherever they need to.

The *Personal Assistant Web Application* provides two forms of displaying the clinical tasks. The first is a timeline in which all the clinical tasks are shown over a chronogram. A timeline of activities has the ability to compress multiple tasks into a single continuity without compromising the succession, and the easy understanding of clinical procedures. The benefits from such a representation include the capacity to sequence events and reduce the potential for overburdening the health care professional. The other available view is a calendar in which the health care professional can visualise the tasks according to the temporal granularity he sees fit, namely week, day, and month. While with the timeline it is easier to detect the starting and ending points of tasks, with the calendar view it is easier to grasp the temporal constraints that bind clinical tasks such as

durations, waiting times and periodicities. Figure 3 shows the same tasks as in the timeline but displayed over a week, where it is possible to verify, for instance, for how long a clinical task should be applied.

Fig. 3. Calendar view of clinical procedures in the CompGuide Personal Assistant Web Application.

The *Health Care Assistant Mobile App* presents a calendar widget which provides the view and methods necessary to display a calendar and schedule events. This calendar allows the navigation through the months and by clicking on a particular date, all the events for that day are shown below on the calendar.

The *Personal Assistant Web Application* and *Health Care Assistant Mobile App* use the CompGuide web services to request all the patient data and clinical tasks, whereby the recommendations are displayed in a calendar of clinical procedures. The fact that all the data is centralised in only one component allows a better tracking of the user actions, greater control over their decisions and get constant supply of clinical recommendations.

4 Execution Examples

The CompGuide CIG execution server and the CompGuide editor for CIG creation and editing was tested using the NCCN Clinical Practice Guideline for Colon Cancer [2]. The representation of the NCCN guideline in the model was supported by using the CompGuide editor plugin. The representation of the NCCN protocol served to verify whether the tool for CIG management was able to correctly assist health care professionals in creating and editing CIGs as well as determine if the CIG execution engine was capable of providing and scheduling the clinical recommendations.

For demonstration purposes, the following expression will be used: “apply medication for neoadjuvant therapy every two weeks for two-three months” extracted from [2]. Through the analysis of the expression, it is possible to identify the *Action* to apply medication for neoadjuvant therapy, the periodicity

value of two with a temporal unit of week, a minimum duration value of two, a maximum duration value of three, and the respective temporal unit of month.

To represent this expression, the CompGuide plugin editor facilitated the acquisition and the representation of a CPG network of tasks, by providing information step-by-step on which fields are required for the definition of each task and associated constraints. This tool provides features such as the capacity to re-utilise the knowledge contained in the CompGuide ontology as shown in Fig. 4 and provides the verification of the required data, meaning that the plugin verifies if the required data is correctly inserted while proceeding with the creation/editing process. Finally, the CompGuide plugin editor confirms the newly created CPG and Plan instances (as seen in the Protégé Entities tab) with the version and dates managed by the wizard. After this process, the CompGuide editor plugin sends the CPG proposal to the CompGuide CIG execution server. CompGuide CIG execution server will save the owl file in the guidelines repository, so later the information can be accessed and retrieved by the CIG execution engine.

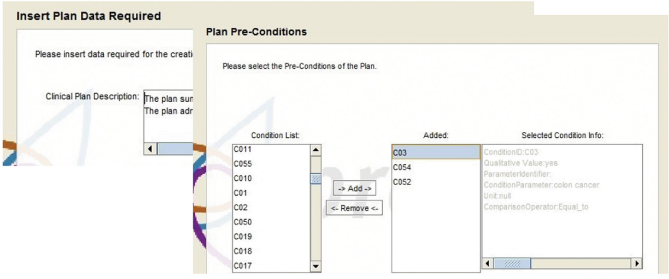

Fig. 4. Creation of the global colon cancer plan.

Once these recommendations are retrieved by the CIG execution engine, their constraints (in this case, their temporal constraints) are interpreted and mapped onto the different views made available by the web application or the mobile application. For the given example the execution engine would recommend the execution of the task, divided in a set of events, with the specified frequency at least for two months and at most for three. So, the application will unfold the recommendation in multiple events and register them in the timeline. As such, the result would be six new calendar events from the start date of the task execution up to three months. The first and second events start on the 18^{th} of July and 1^{st} of August, as shown in the Fig. 5(a). The third and fourth start on the 15^{th} and 29^{th} of August, as depicted in Fig. 5(b). Finally, the fifth and sixth events start on 12^{th} and 26^{th} of September, as shown in Fig. 5(c).

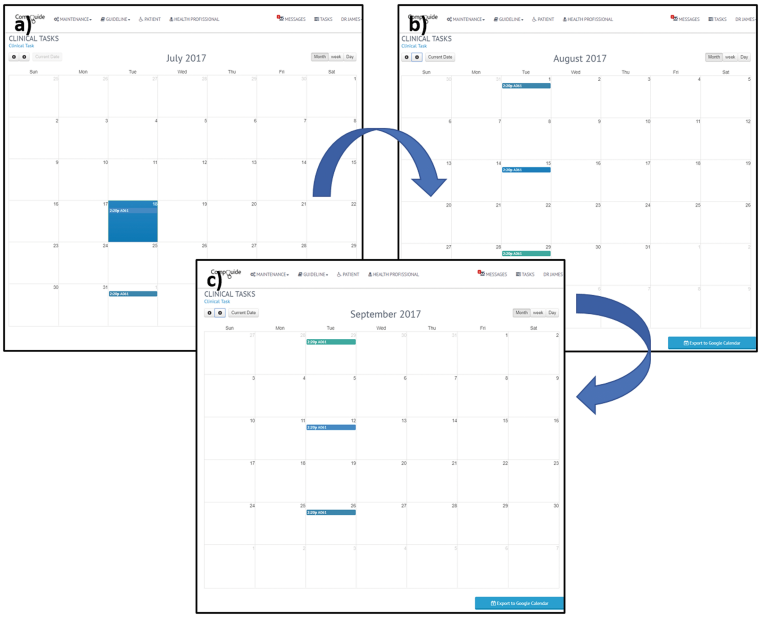

Fig. 5. Execution of a clinical task from the given expression, as seen in the Personal Assistant Web Application. Figures (a), (b), and (c) show different consecutive execution times.

5 Conclusions and Future Work

Several CIG representation languages and CIG tools were created to help health professionals to manage and execute CIGs. However, they lack in providing mechanisms to allow a workflow from creation to execution. The system presented herein aims to increase the effectiveness, interactivity and pervasiveness of CDSSs by providing a new method to integrate CPG advice in a clinical setting and providing a set of functionalities that address the above-mentioned limitation. In addition to decision support functionalities, common to other CIG systems, the CompGuide system allows the development of additional features such as scheduling, alert features and capacity to manage and share CIGs that aims to assist the health care professional in their duties. Furthermore, this system provides a unified workflow for management and execution of CPGs that offers a better possibility of affecting clinical behaviour since it presents features which other approaches do not.

Although the system possesses a set of features that promote the use of the best clinical practices, we recognise that it requires a real-life evaluation. So, we intend to do experiments with physicians by using the system and its implementations to obtain advice about their respective patients. Furthermore, this will allow the comparison of recommendations provided by the system to those the health care professional would usually issue.

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.

References

1. Beard, N., Campbell, J.R., Huff, S.M., Leon, M., Mansfield, J.G., Mays, E., McClay, J.C., Mohr, D.N., Musen, M.A., O'Brien, D., et al.: Standards-based sharable active guideline environment (SAGE) A Project to Develop a Universal Framework for Encoding and Disseminating Electronic Clinical Practice Guidelines. In: AMIA (2002)
2. Benson, A., Bekaii-Saab, T., Chan, E., Chen, Y.J., Choti, M., Cooper, H., Engstrom, P.: NCCN Clinical Practice Guideline in Oncology Colon Cancer. Technical Report, National Comprehensive Cancer Network (2013). http://www.nccn.org/professionals/physician_gls/f_guidelines.asp
3. Gonçalves, F., Oliveira, T., Neves, J., Novais, P.: CompGuide: Acquisition and editing of computer-interpretable guidelines. In: World Conference on Information Systems and Technologies. pp. 257–266. Springer, Cham (2017)
4. Leong, T.Y., Kaiser, K., Miksch, S.: Free and open source enabling technologies for patient-centric, guideline-based clinical decision support: a survey. Yearbook of medical informatics, p. 74 (2007)
5. Musen, M.A.: The protégé project: a look back and a look forward. AI Matters **1**(4), 4–12 (2015)
6. Novais, P., Costa, R., Carneiro, D., Neves, J.: Inter-organization cooperation for ambient assisted living. JAISE **2**(2), 179–195 (2010)
7. Noy, N.F., Crubézy, M., Fergerson, R.W., Knublauch, H., Tu, S.W., Vendetti, J., Musen, M.A., et al.: Protege-2000: an open-source ontology-development and knowledge-acquisition environment. In: Annual Symposium proceedings. AMIA Symposium, vol. 2003, p. 953 (2003)
8. Oliveira, T., Novais, P., Neves, J.: Development and implementation of clinical guidelines: an artificial intelligence perspective. Artif. Intell. Rev. **424**, 999–1027 (2014)
9. Oliveira, T., Silva, A., Neves, J., Novais, P.: Decision support provided by a temporally oriented health care assistant. J. Med. Syst. **41**(1), 1–13 (2017). https://doi.org/10.1007/s10916-016-0655-6
10. Rubin, D.L., Noy, N.F., Musen, M.A.: Protege: a tool for managing and using terminology in radiology applications. J. Digit. Imaging **20**(1), 34–46 (2007)
11. Shahar, Y., Miksch, S., Johnson, P.: The asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines. Artif. Intell. Med. **14**(1–2), 29–51 (1998)
12. Sonnenberg, F., Hagerty, C.: Computer-interpretable clinical practice guidelines. Where are we and where are we going, 145–158 (2006)
13. Terenziani, P., Montani, S., Bottrighi, A., Torchio, M., Molino, G., Correndo, G.: The GLARE approach to clinical guidelines: main features. Stud. Health Technol. Inform. **101**(3), 162–166 (2004)
14. Votruba, P.: Structured knowledge acquisition for asbru. na (2003)
15. 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. J. Biomed. Inform. **37**(5), 305–318 (2004)
16. Young, O., Shahar, Y.: The spock system: developing a runtime application engine for hybrid-asbru guidelines. Artif. Intell. Rev. **3581**(1), 166–170 (2005)