CompGuide: Acquisition and Editing of Computer-Interpretable Guidelines

Filipe Gonçalves[1], Tiago Oliveira[2], José Neves[2], and Paulo Novais[2]

Algoritmi Research Centre/Department of Informatics, University of Minho
Braga, Portugal
[1]pg25309@alunos.uminho.pt, [2]{toliveira,jneves,pjon}@di.uminho.pt

Abstract. The formalization of Clinical Practice Guidelines (CPGs) as Computer-Interpretable Guidelines (CIGs) has the potential to positively influence the behaviour of health practitioners by being available at the point and time of care. Existing tools for acquiring and editing CIGs for automatic interpretation present limitations in their ease of use and the support they offer to a CIG encoder. Besides characterizing these limitations and identifying improvements to include in future tools, this work describes the CompGuide Editor, a Protégé tool for the management of CIGs that guides a user throughout the several steps of CIG encoding, without requiring the user to have programming knowledge, and through the use of interfaces that are simple and intuitive.

1 Introduction

The deployment of Clinical Practice Guidelines (CPGs) in Clinical Decision Support Systems (CDSSs) for daily use has several obstacles, namely the development of systems that are able to provide the level of interaction with the user required for the task, handling data entry points and the multiple control structures in CPGs, and the development of interfaces for conveying recommendations in an efficient way [5,13]. These issues have been tackled in previous works through 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 [7,11]. However, even before considering the operationalization of CPGs, there is the issue of knowledge acquisition. Transforming CPGs into computer algorithms from their text versions is a difficult task. These versions were not originally designed to be interpretable by computers and, in some cases, contain complex and intricate instructions, involving several parameters, that are difficult to translate into efficient algorithms [4]. This led to the development of different Computer-Interpretable Guideline (CIG) models and tools by different research groups, covering a wide range of clinical situations [6,13]. A model (here used as a synonym for language) aims to provide a structure for the correct formalization of a narrative CPG as a CIG that would be the basis of a CDSS, but by itself it is not sufficient, it is still necessary to guide CIG design, according to a selected model, in order to ensure a correct syntax and disposition of knowledge elements. While

it is true that there are several tools for helping users to create effective CIGs, they have not, to the extent of our knowledge, progressed beyond the level of academic research projects. Furthermore, they reveal limitations in guiding the user throughout the CIG acquisition process, the intelligibility of their interface, the visualization of guideline knowledge elements and attached information, and the sharing of CIGs.

The present work discloses a CIG editing tool, the CompGuide Editor. The underlying model for CIGs used is the CompGuide ontology [9], which is based on Web Ontology Language (OWL). The CompGuide ontology presents a formalisation of guidelines as linked lists of tasks, thus following the Task Network Model (TNM), representing CPGs as workflows. The CompGuide Editor performs the role of managing the creation and editing of CIGs as a user-friendly tool to represent clinical guidelines in a specific model, able to address the limitations of the existing applications.

The present paper is organized as follows. Section 2 provides related work on CIG creation and editing tools. Their most prominent aspects and main deficiencies are exposed in this section. Section 3 explains the steps taken in the creation of the CompGuide Editor tool. Finally, section 4 presents conclusions about the work developed so far and future work considerations.

2 Computer-Interpretable Guideline Acquisition and Editing

As already mentioned, there are several tools that support the acquisition and editing, either manually or semi-automatically, of CIG knowledge elements, taxonomies, and other organization levels of these machine-readable formats.

2.1 Features for Tool Assessment

This section is dedicated to defining features for the comparison of CIG acquisition and editing tools. The comparative features were selected as a means to analyse and evaluate CIG platforms, based on user experience [3]. They consist of the following:

- **Graphical Guidelines View**: a graphical representation (tree, node-link, network diagrams) of parts of or a full CIG workflow. The arrangement of the representations within a drawing helps the user to understand the workflow, identify relevant points of the guideline, and manipulate knowledge elements;
- **Organization**: this feature is related with how easy the tool is to understand, determined by its structure and the way in which its functionalities are made available, whether they are placed correctly, under the right menu;
- **Simplicity**: this feature conveys the ease of access to the functionalities of the tool. Complexity leads to confusion in the use of the tool, leading many users to abandon it;

- **Automation**: when creating or editing new instances, the user should only implement the most relevant knowledge elements, with the rest being automatically completed by the platform;
- **Drag-and-Drop**: the ability to drag-and-drop instances in the Graphical Guideline View and filter the workflow of the CIG with the help of graphical type links;
- **Web/Local Repository**: the possibility to save or load CIGs either locally or in a cloud repository;

2.2 Assessment of Existing Tools

The first tool to be analysed was the Protégé Desktop, an open source ontology development and knowledge acquisition environment developed by Stanford Medical Informatics [8]. It provides a platform which can be extended with graphical widgets for tables, diagrams, and animation components. The tool is used to author guidelines in various models, including the Guideline Interchange Format (GLIF) [12] and PRODIGY [14].

The SAGE Workbench is a complete, self-contained environment that uses the Shareable Active Guideline Environment (SAGE) model [1]. The SAGE Workbench provides a knowledge authoring tool based on Protégé. The user interface for the SAGE Workbench is organized as a number of tabs for: the navigation of frames that are directly and indirectly referenced from a selected instance in a tree structure, expression of integrity constraints about a knowledge base in *Protégé axiom language*, search of terms in a medical terminology service, and so forth. The Tallis tool is a relatively recent (when compared to other tools) Java implementation of PROforma-based authoring and execution developed by the Cancer Research UK [17]. *Tallis* is based on a later version of the PROforma model [16]. It is, in fact, a suite of tools in which the main component is a desktop graphical editor for authoring guidelines. Workflows are displayed in Tallis both in a network view and in a tree view. GEM Cutter is a simple XML editor that facilitates the markup of CPG texts, and therefore supports the conversion of a guideline document into the GEM format and publication in a cross-platform manner [15]. The main window of the tool comprises three panels, a menu bar, and a button bar. The CPG text is loaded into the leftmost panel. The middle panel contains an expandable tree view of the GEM hierarchy. The rightmost panel shows information regarding the knowledge element selected at the time. Finally, 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 [18]. There is a topological view to observe the relationships between the different plans and a temporal view that shows the temporal constraints of plans. Table 1 shows a comparison between these tools, by using the comparative features explained earlier. Important to say that this comparison does not include plug-ins that can be applied to these platforms in order to add new features. We can conclude that some features are absent from the existing tools and should be developed in the CompGuide Editor tool to

improve the user experience. Although all the tools possess the basic requirement of allowing the instantiation and editing of CIG elements, they are mainly focused on the proper functioning of the tool and less on the appearance or ease of management.

Most tools (the exception is GEM Cutter) present some form of graphical view of the workflow. Whether it is through a network structure or a tree structure, it is possible to have a general view of the instances that build a CIG and then focus on particular elements. Although the analysed tools are clearly well organized, they lack simplicity. Taking Protégé Desktop and SAGE Workbench as examples, despite their having many useful features available, the amount of menus they display is significant, which makes the user lose a lot of time trying to understand the different functionalities. As such, simplicity is a feature that should be prioritized. As for the drag-and-drop feature, it is absent from existing tools, but an important element in modern applications, enabling the management of a CIG visual layout. One of the features that new tools should present is the ability to automatically fill in data that is not so important, allowing the user to focus only on the data relevant to the instance he is currently creating, leaving the less important details to be handled by the system. This feature includes the automatic creation and filling of mandatory data fields for an instance, such as the date of creation of a CPG or its version, the relationships expressing the connection of a clinical task to a subsequent task, and so forth. This kind of automation is not present in none of the analysed CIG tools. Another important feature is the ability to import or export CIGs stored locally or in a cloud. We must give relevance to this feature, as most of the information is currently held in clouds, giving the possibility to the user to access all this data anywhere, any time, and share it with other members of a CIG development team. Only Protégé Desktop and SAGE Workbench show this feature.

Table 1. Comparison of tools for the acquisition and editing of CIGs. The x shows that the tool possesses the feature.

Feature/Platform	*Protégé Desktop*	*SAGE Workbench*	*Tallis*	*GEM Cutter*	*Asbru View*
Graphical Guidelines View	x	x	x		x
Organization	x	x	x	x	x
Simplicity			x	x	x
Automation					
Drag-and-Drop					
Local Repository	x	x	x	x	x
Web Repository	x	x			

3 The CompGuide Editor Tool

The CompGuide Editor was developed as a Protégé Desktop plug-in given the need to create software capable of implementing all the features offered by this application, more specifically the functionality of managing the data of an ontology through the use of a graphical interface, along with the development of new features capable of solving the limitations in existing projects. Another advantage that came from using Protégé was the ability to implement extra features from other plug-ins in a simple way. Since OWL is one of the ontology languages supported by Protégé and, at the same time, the underlying language of the CompGuide ontology, using this application as the basis for the editor was the logical decision to make.

3.1 Characteristics of the CompGuide Ontology

In the CompGuide model all the knowledge elements of CPGs are represented as different tasks[9]. The classes that enable this are *Plan*, *Action*, *Question* and *Decision*. 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 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. Time is a crucial dimension in the representation of clinical procedures. The temporal constraints in CompGuide 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. CompGuide provides constructors to express durations, periodicities, and waiting times of tasks. Furthermore, it is also possible to define temporal constraints for conditions about the state of a patient [11]. A distinctive feature of CompGuide, when compared to other CIG models is that it does not require any proficiency in a programming language in order to define constraints. Instead, all is done with the basic elements of an ontology: classes, instances, and properties. It was demonstrated that this CIG ontology was able to provide sufficiently expressive constructors for CPGs from different medical specialities and categories [10].

3.2 Development of the Plug-in and Main Features

Due to compatibility issues and the available support, the CompGuide Editor was developed for Protégé 4, using various Java APIs such as OWL extension APIs, the Protégé OWL API, the Protégé API, and the Jena API.

The main challenge in the development of the editor is the little knowledge users may have about the terms of the underlying CompGuide ontology used in the functionalities, which implies longer learning curves in becoming acquainted with the concepts and learning how to create ontology-enabled representations. To tackle this challenge the CompGuide Editor implements a set of wizards, namely the Create Wizard, the Edit Wizard, and the Delete Wizard. These

are software assistants that guide de user step-by-step in the task they want to perform with a sequence of dialogue boxes. Figure 1 shows one of these boxes for the Create Wizard, regarding the creation of a *Plan*. Given the richness of knowledge elements in the CompGuide ontology, the wizards are essential in order to guarantee that the user has access to all the needed constructs. This is an improvement with regards to existing tools, where such a set-up feature is not available. The wizards guarantee the intended simplicity for the editor as they insure that the user has an easy access to the functionalities he needs for the task he is carrying out. Other existing Protégé-based tools, namely Protégé Desktop and the SAGE Workbench, possess complex functionalities, but do not provide easy access to them.

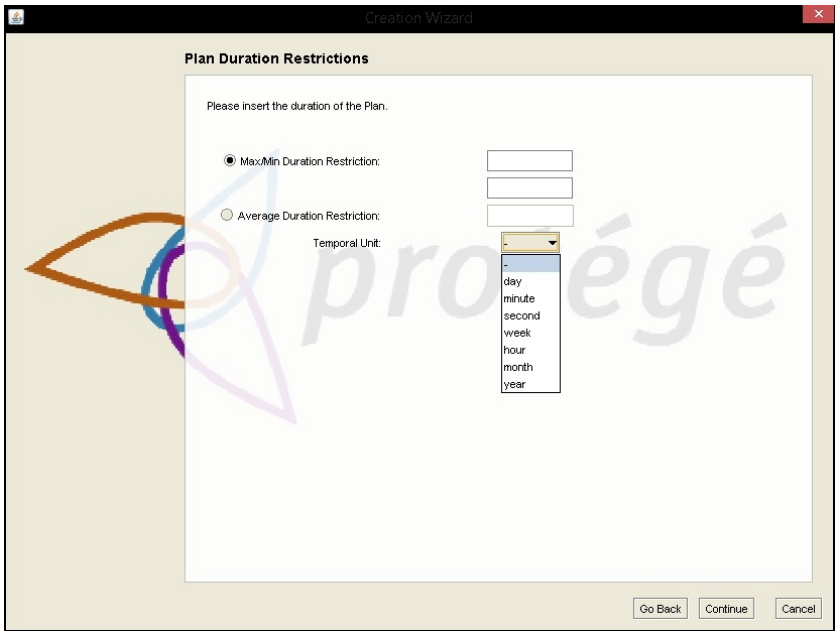

Fig. 1. CompGuide Editor Create Plan Wizard.

Figure 2 shows the main interface of the CompGuide Editor, which is viewed as a tab in Protégé. The interface consists of two main views. The OntoGraf view provides a 2D dynamic graphical representation of the ontology, while the CGuide Wizard options view (bottom view) has the set of features to manage the CIG plus the options to download/upload the CompGuide ontology file. In addition to the graphical view of the CIG, it is possible to see CIG elements in a list, organized by the class to which they belong, through the Individuals by Type view. The number shown in front of the the name of the class represents the number of individuals that exist in that OWL class. When accessing the CIG

management features, a new window will appear, in which a simple and intuitive step-by-step procedure will start. In order to complete this process, the user must follow the instructions of the corresponding wizard and insert all the required data. In the end, the CompGuide Editor tool will automatically insert all changes in the ontology. Figure 1 shows one of the steps in the creation of an instance in the CPG, in this case the definition of a temporal constraint, a duration, for a *Plan*. The wizards handle all the data structures and automatically fill in the names of instances, dates, versions and the connections between instances. For instance, a CPG instance is always associated with an instance of *Scope*, which specifies the range of the CPG. When the CPG instance is created, the skeleton for the *Scope* instance is immediately and automatically generated.

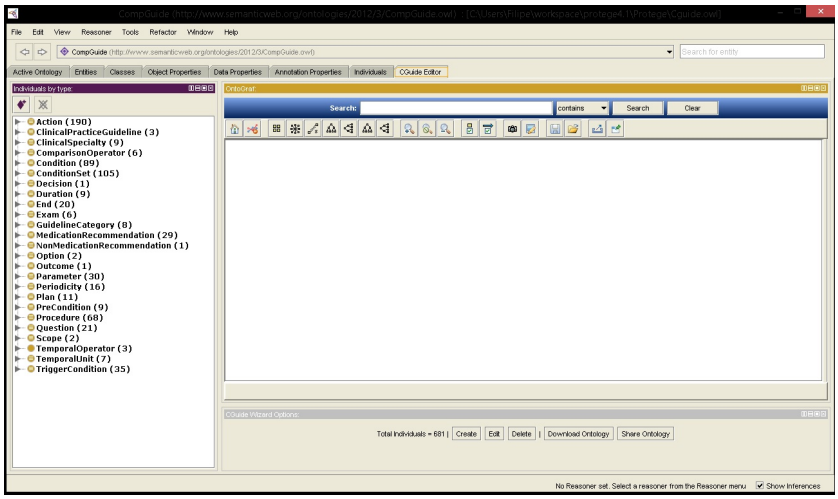

Fig. 2. CompGuide Editor main interface.

The OntoGraf view also allows the user, through the drag-and-drop feature, to manipulate the graphical representation of a CPG, such as the one shown in Figure 3. These changes include the disposition of the nodes representing instances and the selective expansion of nodes in order to view their object and data properties. It is also possible to glance at a the information of a node by placing the mouse on top of it, as seen in Figure 3.

The interactions of users with this tool are represented in Figure 4. While health care professionals are responsible for the creation, modification or deletion of clinical steps or aspects in a CIG file, the administrator (or admin) has the responsibility to maintain the latest CIG version in the Git repository, which can be downloaded and used by the health care professional in the CompGuide Editor. Once the user finishes editing the file, he may upload it to a file repository

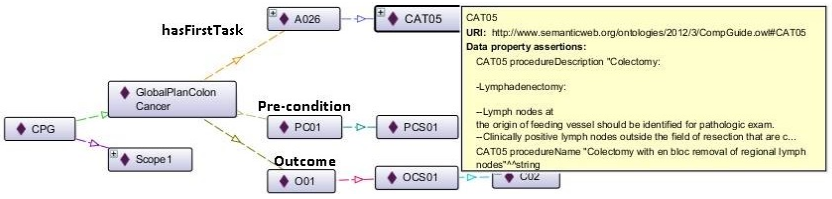

Fig. 3. Graph containing instances of a *Plan* for the management of colon cancer.

for the admin to check. If the file is considered to be valid, it is placed in the Git repository for further use in the editor.

3.3 Assessment remarks

In order to verify the usability of the CompGuide Editor, a CPG of the National Comprehensive Cancer Network (NCCN) for Colon Cancer [2] was fully represented in the CompGuide ontology. The CPG is used for the diagnosis and management of colon cancer and, thus, contains numerous clinical tasks with complex relationships. The process of representing the guideline resulted in an*owl* file with 680 instances, out of which 223 were task instances. The graph of Figure 3 shows the initial portion of the CPG with its main *Plan* and corresponding first task. Since this is a lengthy CPG, the CompGuide facilitated its acquisition by providing information step-by-step on which fields are required for the definition of each task and associated constraints. This was particularly useful in the definition of the procedural logic of the CPG, the sequence of clinical tasks, and splitting points in he CPG workflow, where it is necessary to choose one from multiple alternative tasks. The graphical view was particularly useful in the visualization of this later aspect, allowing a rapid comprehension of the CPG workflow by selectively expanding and shrinking parts of the graph. These were the aspects in which CompGuide proved to be more useful.

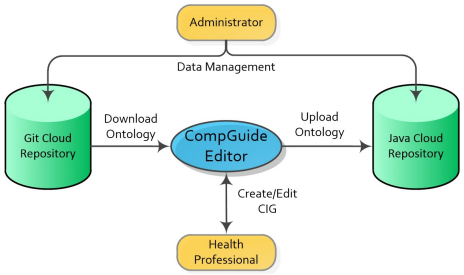

Fig. 4. Actors in the CIG acquisition and editing system involving the CompGuide Editor.

4 Conclusions and Future Work

There are several languages and tools helping final users and system developers in creating good and effective CIGs, such as Protégé Desktop, SAGE Workbench, Tallis, GEM Cutter, Asbru View, among others. Yet most of these platforms lack some important features, leaving place for improvements. The features identified as lacking were simplicity, automation, drag-and-drop functionalities in the graphical views, and CIG sharing functionalities. By studying these tools, a plug-in for Protégé Desktop application was developed, with the capacity to manage and share CIGs formalized in the CompGuide ontology. Besides the basic features, also present in the above-mentioned tools, the idealized CompGuide Editor provides a set of functionalities that make it more complete then its existing counterparts. The simplicity feature is provided by the set of wizards implemented in the tool, which guide de user step-by-step, easing the understanding of functions and knowledge elements. These assistants also carry the task of managing data that is less important from the viewpoint of the user, but vital to internal consistency of the CPG. The graphical view in the tool confers a greater freedom in the manipulation and visualization of knowledge elements. The cloud repository and the workflow involving the download and upload of CIG files ensures that the CPGs can be shared among interested parties. This work focuses on the requirements of tools for the acquisition and editing of CIGs. One of the core characteristics of the proposed tool, the CompGuide Editor, is that it does not require programming skills for the definition of CIG elements.

Although the tool disclosed herein possesses a set of desirable features that makes it, at least from a theoretical analysis, an improvement over existing tools of the same type, it is necessary to conduct a usability study to determine if that is in fact the case. As such, this is the next step regarding the development of the CompGuide Editor. The tentative procedure to make this assessment is to select a tool from the set of reviewed tools and propose the representation of the same CPG to a sample of subjects, with that tool and the one presented herein. After the representation the assessment can be made through questionnaires.

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.

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). 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. Tech. rep., National Comprehensive Cancer Network (2013), `http://www.nccn.org/professionals/physician{_}gls/f{_}guidelines.asp`

3. Bott, R.: Summary of the Guideline Workbenches Evaluation. Igarss 2014 (1), 1–5 (2014)
4. Chim, JCS and Cheung, NT and Fung, H and Wong, K.: Electronic clinical practice guidelines: current status and future prospects in Hong Kong. Hong Kong Medical Journal 9(4), 299—-301 (2003)
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 (may 2004)
6. Isern, D., Moreno, A.: Computer-based execution of clinical guidelines: a review. International journal of medical informatics 77(12), 787–808 (2008)
7. Novais, P., Costa, R., Carneiro, D., Neves, J.: Inter-organization cooperation for ambient assisted living. JAISE 2(2), 179–195 (2010)
8. 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: AMIA Annu Symp Proc. vol. 953, p. 953 (2003)
9. Oliveira, T., Novais, P., Neves, J.: Development and implementation of clinical guidelines: An artificial intelligence perspective. Artificial Intelligence Review pp. 999–1027 (2014)
10. Oliveira, T., Novais, P., Neves, J.: Assessing an Ontology for the Representation of Clinical Protocols in Decision Support Systems. In: Bajo, J., Hernández, J.Z., Mathieu, P., Campbell, A., Fernández-Caballero, A., Moreno, M.N., Julián, V., Alonso-Betanzos, A., Jiménez-López, M.D., Botti, V. (eds.) Advances in Intelligent Systems and Computing, vol. 372, pp. 47–54. Springer International Publishing (2015)
11. Oliveira, T., Silva, A., Neves, J., Novais, P.: Decision Support Provided by a Temporally Oriented Health Care Assistant. Journal of Medical Systems 41(1), 13 (2016)
12. Peleg, M., Boxwala, a.a., Ogunyemi, O., Zeng, Q., Tu, S., Lacson, R., Bernstam, E., Ash, N., Mork, P., Ohno-Machado, L., Shortliffe, E.H., Greenes, R.a.: GLIF3: the evolution of a guideline representation format. In: Proceedings / AMIA ... Annual Symposium. AMIA Symposium. pp. 645–649. American Medical Informatics Association (2000)
13. Peleg, M.: Computer-interpretable clinical guidelines: A methodological review. Journal of Biomedical Informatics 46(4), 744–763 (jun 2013)
14. Purves, I.N., Sugden, B., Booth, N., Sowerby, M.: The PRODIGY project–the iterative development of the release one model. Proceedings / AMIA ... Annual Symposium. AMIA Symposium pp. 359–63 (jan 1999)
15. Shiffman, R.N., Agrawal, A., Deshpande, A.M., Gershkovich, P.: An approach to guideline implementation with gem. Studies in health technology and informatics (1), 271–275 (2001)
16. Steele, R., Primer, F.J.T.P.: Introduction to proforma language and software with worked examples. Tech. rep., Technical report. London, UK: Advanced Computation Laboratory, Cancer Research (2002)
17. Sutton, D.R., Fox, J.: The syntax and semantics of the proforma guideline modeling language. Journal of the American Medical Informatics Association 10(5), 433–443 (2003)
18. Votruba, P.: Structured knowledge acquisition for asbru. na (2003)