University of Minho

School of Engineering

Tiago Campelo Vilas Boas

Development and Implementation of Automatized Clinical Practice Guidelines

Master's Dissertation

Master's Degree in Computer Engineering

Work done under the supervision of

Paulo Jorge Freitas de Oliveira Novais

Tiago José Martins Oliveira

November 2013

Acknowledgments

This project was a surprise for me, it started as a simple idea behind an interesting project, but sooner than later it became clear the size of the task ahead of me, and the amount of hours, dedication and coding I had to give to achieve only a part of the goal.

So I would like to thank my supervisors, Tiago Oliveira and Paulo Novais for the amount of questions they so willingly answered to allow me to advance with the project, my parents, brothers, nephews, and also my friends for continuing to be so during these long months developing and writing this dissertation.

Título

Desenvolvimento e Implementação Automatizados de Recomendações Clínicas

Resumo

Atualmente no setor da saúde onde, a crescente necessidade de estandardizar as práticas médicas e promover a melhoria das práticas clínicas de forma a reduzir os custos exige uma solução que permita que estes objetivos sejam mais facilmente atingidos. Para o efeito, a solução que mais desperta o interesse atualmente é a utilização de Recomendações de Práticas Clínicas.

Recomendações Clínicas oferecem, na forma de documentos longos, conselhos de boas práticas médicas em circunstâncias específicas. Contudo, por vezes os conceitos existentes nestes documentos são vagos e inconclusivos. Para além disso, o seu tamanho e formato desestruturado torna os difíceis de consultar e de estarem disponíveis no momento do ato médico. Daqui surge a ideia de representar estas recomendações computacionalmente, isto é, recorrer a *Computer-Interpretable Guidelines*, que são representações estruturadas de recomendações clínicas, que podem ser interpretadas e executadas por máquinas.

O que se pretende assim nesta dissertação, é o desenvolvimento de uma aplicação web que permita a criação e implementação de forma simples e estruturada de Recomendações Clínicas, permitindo aos profissionais de saúde facilmente criá-las, aplicá-las e modificá-las.

A aplicação é capaz de, a partir do modelo de representação de *guidelines,* permitir a um utilizador aplicar uma *guideline* representada e executá-la. A aplicação oferece todos os recursos necessários para consultar recomendações clínicas, e armazena toda a informação processada e gerada pela aplicação numa base de dados que está localizada junto com o servidor principal. Conta ainda com uma aplicação web que permite a interface com o utilizador tanto para executar *guidelines* como para gerir utilizadores.

Title

Development and Implementation of Automatized Clinical Practice Guidelines

Abstract

Currently in the health sector, the growing need to standardize medical practice, improve clinical practices and reduce costs call forth the need to find a solution that allows these objectives to be achieved more easily. To achieve this purpose, the solution that seems most promising is the use of Clinical Practice Guidelines.

Clinical Practice Guidelines offer recommendations in the form of long documents with the most recommended medical advice in a certain situation. However, sometimes the concepts within these documents are vague and inconclusive. Furthermore, the size and unstructured format of the documents makes them difficult to consult and be available at the time of the medical act. Hence the idea of representing these recommendations computationally, i.e., using Computer-interpretable Guidelines, which are structured representations of Clinical Practice Guidelines that may be interpreted and executed by machines. The aim of this dissertation was the development of a web application that allows the implementation of CPGs, allowing medical professionals to easily create, apply and modify them.

The application manages to take a model of Computer-Interpretable Guidelines and allows the user to apply the guideline represented on that specific model and execute it. The application provides all the resources needed for the simple consultation of guidelines, it also stores all the processed information as well as a data model in a safe database stored along the web server which provides the web site the ability to interface with the user.

Table of Contents

Figure Index

Notation and Terminology

Notation

Throughout the document italic text is used when referring to computer technologies and medical and mathematical terms.

Terminology

AI	Artificial Intelligence
AIM	Artificial Intelligence Medicine
CDSS	Clinical Decision Support Systems
CPG	Clinic Practice Guidelines
CUI	Code Unique Identifier
DeGel	Digital Electronic Guidelines Library
EMR	Electronic Medical Record
GLARE	Guideline Acquisition, Representation and Execution
GLEE	GLIF3 Guideline Execution Engine
GLIF	the Guideline Interchange Format
HeCaSe2	Health Care Services
ICT	Information and Communication Technologies
IT	Information Technologies

JSP	JavaServer Pages
OCL	Object Constraint Language
OWL	Web Ontology Language
SAGE	Standards-based Sharable Active Guideline Environment
SpEM	Specification Execution and Management
TNM	Task Network Model
UMLS	Unified Medical Language System
UN	United Nations
W3C	World Wide Web Consortium
WHO	World Health Organization
XML	eXtensible Markup Language
ISLab	Intelligent Systems Lab

1. Introduction

Clinical practice guidelines (CPGs) are documents that aim to offer health practitioners guiding decisions and criteria regarding diagnosis, management and treatment of a patient. These guides have been widely recognized has one of the most efficient mean to standardize and optimize health care practices and studies show that if the integration of CPGs into the clinical workflow were to increase, they would improve the quality of care and remove unwanted variables, human error and the costs associated with them [1].

Despite clinical guidelines showing incredible potential and despite the efforts to make use of CPGs on a daily basis, their use has not increased due to the fact that CPGs are available through paper documents, sometimes long and difficult to consult, especially during the process of clinical practice. In this format, it is difficult for them to keep up with the fast growth and development of scientific knowledge because they are difficult to update. It became challenging for CPGs to have the impact and improve health care in the way that was expected.

The attention turned into a relatively recent but very successful field at the intersection of computer science, information science and health care, Health Informatics (HI). HI already made huge contributions to health care through Electronic Medical Records, Information Retrieval applied to clinical contents and Decision Support Systems [2].

But CPGs have not gotten much of its attention yet, making them one of the most interesting topics today in HI. Many believe that the solution to implement CPGs in daily use will be found in this field, by developing applications to simply create and execute clinical guidelines[3]. This presents itself has a solution to help CPGs have a stronger presence and impact in the health system.

1.1. Motivation

A patient that trusts his life to health care professionals during any medical procedure expects the best possible treatment available. The health care professionals hold the responsibility of making the best decisions during the clinical process according to each diagnosis and specific case. There is a wide variety of factors that can indeed affect the way practitioners make their decisions and may lead to medical errors, excessive costs and variations in clinical practice. The World Health Organization (WHO) claims that currently in Europe, 6% of the annual spending in health care systems is lost due to errors and corruptions and in the United States of America this percentage rises to 20%. Although some countries loose more than others, there are still plenty of ways to improve efficiency. Some of the reasons, for this, according to WHO [4], are medical errors and the sub-optimal quality of care due to insufficient knowledge or application of protocols and lack of guidance throughout health care practice. Inefficient and inappropriate use of medicines, limited knowledge and bad practices are also one of the main causes for these issues in the current health care systems.

1.1.1. Medical Error

Medical error can be defined as a situation where a health care provider chooses a wrong method of care or chooses the right method but it was executed incorrectly [5].

Occupational stress is the most studied and the main cause of medical error [6], given the negative impact on health care, both for an individual and an organization. Occupational stress is a result of different factors such as working conditions, excessive working hours, ambiguity of responsibilities, inter-personal relationships in the work place and career development.

Health practitioners belong to a professional group that is particularly susceptible to stress. The pressure of dealing with patients face to face has been recognized as an important factor in the origin of emotional exhaustion and psychological breakdown. This can be enhanced by the eventual connection between the professional and the patient, the need to sometimes inflict pain through medical procedures and dealing with severe injuries, physical pain, grief and death in a daily basis.

Contrary to other professions, in health care, the medical error can result in the deterioration of the patient's condition or even death. In turn, this may lead to a spiral of stress. The increase in difficulty and responsibility of each decision and the professional's own stress can affect the center of his attention, the patient.

According to WHO[4], in the countries in the European Union, on average, a medical error occurs in about 8-12% of all hospitalizations. These values, although apparently very low, have a bigger dimension considering the context of health care, where a medical error can bring huge consequences.

To solve these issues, providing information during the medical process that can aid the decision of the professional should ease the emotional, physical and psychological fatigue. Using CPGs has proven useful and the receptivity of this kind of tools by professionals is high. According to several studies around the world, 75% of professionals consider CPGs a useful source of recommendations and 71% agree that CPGs are good teaching tools[7]. Most considered that having them in an application that can be easily consulted should improve the quality of service [8].

1.1.2. Defensive Medicine

Defensive medicine is a common practice for many health care practitioners which consists in the deviation of what is considered good practice in order to avoid lawsuits and complaints from patients and their families [9]. While defensive medicine can be translated into determination and consideration by the practitioner to improve the patient's condition and avoid errors, bad defensive medicine consists in an attempt of the practitioner to maintain professional integrity by avoiding certain procedures and patients of high risk. This practice sometimes happens involuntarily in health care professionals and is normally associated with more experienced professionals. Defensive medicine contributes to the increase of costs of medical procedures, especially with the excessive use of complementary means of diagnosis and the unnecessary use of medicines which themselves represent 20-30% of the global spending in health care[4]. Defensive medicine is motivated by the notion that doing too much can never hurt, which is viewed by professionals as a way to protect themselves from potential lawsuits but leads to wasteful practices [10].

The implementation of easily usable and upgradeable Computer-Interpretable Guidelines (CIGs), CPGs that are translated into machine interpretable language to be read and executed by computers, can reduce defensive medicine by sharing the responsibility associated with the practitioner and the practice being executed, guiding him throughout the whole process avoiding the use of both excessive medical procedures and medicine prescriptions.

1.1.3. Variation in Practice

Because health care professionals learn from other professionals and medical literature, this leads to wide variations in medical practice from what is considered best practice [11].

Variation in medicine has complex associations, experts vary in their opinions, which confuses physicians in determining the best practice. This combined with medical complexity and the rapid evolution on medical knowledge results in variations in clinical practice. These variations associated with an individual's level of stress, distraction and time constrains can lead to medical error, despite the dedication, effort and intelligence.

To reduce variation is essential to improve health care, and CIGs may be the solution because by definition they are tools for standardizing the medical procedures and removing subjective perspectives.

1.2. Scope

1.2.1. Ubiquitous Computing

We currently live in an age distinguished by the wealth of knowledge and innovation and the health-care sector is one that can greatly benefits from such innovations. The progress in computing during the last 40 years has made information systems indispensable in everyday life. Also, the improvements in micro-processors have pushed us to smaller and more powerful devices, enabling computers to be rooted in so many aspects of our environment[12].

The health care industry is an information intensive business in which Information Technologies (IT) play a key role. New information technologies appear in the health sector once their stability, reliability and usability have been verified. As clinicians adapt to the use computers in their healthcare practices, the focus shifts from financial matters to the realm of patient care. By focusing on using computers to improve health instead of just reducing costs. Further exponential growth can be expected as the healthcare industry pursues an electronic health record system enabled with mobile technology for sharing information and ubiquitous computing. The healthcare sector has developed its prototype of the practice that is performed by individual clinicians or surgical team

[12]. As communication between doctors becomes more significant and the response to the clinical inquiry is becoming the key issue, the access of data anywhere/anytime in a clinical environment is considered an important aspect of the quality of care. With unlimited access and mobile computing, the task of information management can be freed from the constraints of distance and poor communication.

With the ubiquity of computing, on-time decision support will be more practical in healthcare applications. The future of the clinical decision support system depends on progress made in the programs and in reducing implementation barriers. Often, the tasks surrounding information management are delayed in order to receive input or decisions from key stakeholders. Ubiquitous computing lessens not only the physical distance but also access time, providing more opportunities to critical decision makers at any time.

Through ubiquitous computing, clinicians can access clinical data readily and receive specialized opinions in a timely manner. The total aspect of care will be further perfected and the relationship between patients and medical staff will improve. Clinicians can become continuously aware of their patients' needs and provide more natural and powerful means of access to their services.

1.2.2. Artificial Intelligence in Medicine

In the first decade of Artificial Intelligence in Medicine (AIM), most research systems were developed to assist clinicians in the process of diagnosis, typically with the intention that it would be used during a clinical encounter with a patient. Most of these early systems did not develop further than the research laboratory, partly because they did not gain sufficient support from clinicians to permit their routine introduction[13].

It is clear that some of the psychological basis for developing this type of support is now considered less compelling, given that situation assessment seems to be a bigger issue than diagnostic formulation. Some of these systems have continued to develop, however, and have transformed in part into educational systems.

The idea of autonomous systems faded and began to shift into systems that can be deeply integrated into an electronic medical record system. Indeed, such integration reduces the barriers to using such a system, by crafting them more closely into clinical working processes, rather than expecting workers to create new processes to use them. Making these systems more or of a guide that makes suggestions and reminders with the objective to make the practitioner's decisions and procedures easier and more reliable.

Learning is an essential characteristic of an intelligent being. Consequently one of the ambitions of Artificial Intelligence is to be able to develop computers that can learn from experience creating a sub-field of Artificial Intelligence (AI) called machine learning. These systems look at raw data and then attempt to hypothesize relationships within the data, and newer learning systems are able to produce quite complex characterizations of those relationships. In other words they attempt to discover humanly understandable concepts[3].

Machine learning has a potential role to play in the development of clinical guidelines. It is often the case that there are several alternate treatments for a given condition, with slightly different outcomes. It may not be clear however, what features of one particular treatment method are responsible for the better results. If databases are kept of the outcomes of competing treatments, then machine learning systems can be used to identify features that are responsible for different outcomes.

The storing of sensible data that can be collected such as observations of the patient, medication prescriptions to a user and any other relevant information, which by combining with different users, from different ages and sexes can offer valuable information about the efficiency of a given guideline, treatment or medication.

1.3. Objectives

The theme of this dissertation project is *Development and Implementation of Automatized Clinical Practice Guidelines*. The main objective is to develop a ubiquitous and cross-platform system to simply create and execute CIGs, while at the same time creating a global data base of data collected during the execution. The application should offer recommendations based of the current state of the patient.

It will offer an easy to use application for health care practitioners, with the potential to reduce medical error, defensive medicine and medical practice variations, by promoting and stimulating the use of CPGs. This would contribute to the improvement of health care systems and health care in general.

The main objectives of the dissertation project can be resumed as:

- Develop a model of data to store patient data as a medical record and to keep information from previous executions of a guideline;
- Develop a creation mechanism for CPGs:
- Develop an execution engine for CPGs;
- Develop functionalities that enable the application to offer recommendations associated with the collected information;

1.4. Methodology

This project adopted a methodology that will oversee the course of the work: the action-research methodology. This process was triggered by the formulation of a hypothesis that responds to a given challenge.

Further in the report, a compilation of the information relevant to contextualize the developed work will be made along with the respective conclusions to evaluate the results. The project unfolded through the following steps and tasks:

- Definition of the Problem and Characteristics;
- Constant update of the State of the Art;
- Development of a functional prototype;
- Testing the prototype;
- Diffusion of the knowledge obtained to the scientific community.

2. Clinical Practice Guideline Execution

2.1. eHealth

The term eHealth was apparently first used by industry leaders and marketing people rather than academics. The use of *e-words* such as e-commerce, e-business, e-solution, and so on, has the objective to convey promise, principle, and hype to the health arena and to give account of the new possibilities the Internet offers the area of health care.

According to Eysenbach [14], there are ten main objectives of eHealth:

- One of the purposes of eHealth is to increase efficiency in health care, thereby decreasing costs. One of the ways to achieve this is by avoiding duplicative or unnecessary diagnostic or therapeutic interventions;
- Enhancing quality of care. Increasing efficiency involves not only reducing costs, but at the same time improving quality. EHealth may enhance the quality of health by allowing the comparison between different health care providers;
- EHealth contributes for reducing variations, allowing the implementation of CIGs and contributing for a more evidence-based medicine;
- Empowerment of consumers and patients by making Electronic Medical Records (EMRs) available for everyone, opening new paths for patient based medicine;
- Encouragement of new relationships between patients and health professionals, towards a partnership where decisions are made in a shared manner;
- Education of physicians can be made through online resources such as CIGs as well as health prevention actions and education for patients;
- Enabling the sharing of information and communication in a standard way between health care establishments and institutions;
- Extending the scope of the conventional borders, by removing the space barriers, allowing patients to receive health care remotely;

- EHealth introduces new forms of patient-physician interaction and poses new challenges and threats such as online professional practice, informed consent, privacy and equity issues.
- Make health care more equitable is one of the promises of eHealth, but at the same time there is a considerable threat that eHealth may deepen the gap between the *haves* and *have-nots*. People who do not have the money, skills, and access to computers and networks, cannot use computers effectively. As a result, these patients (which would actually benefit the most from health information) are those who are the least likely to benefit from advances in information technology, unless political measures assure equitable access for all.

EHealth should also aim to be simple, fun and exciting to be able to grow and improve health-care. Health concerns all of us, and Information and Communication Technologies (ICT) are increasingly providing us the tools needed to enable solutions for improving health care systems, benefiting patients, practitioners and institutions.

EHealth has already improved health conditions in rural or remote areas and increased the development of health care systems in developing countries. The implementation of these particular systems is a challenge shared by the United Nations (UN) agencies and health authorities at international and local levels. Efforts over the last years have shown there is a clear need for guidance on an approach and methodology which holds the most promise of success[15].

One example was the handling of the aftermath of the tsunami in Indonesia in 2004 where the health responsibilities were coordinated by the WHO. Using ICT's facilities available in the situation room, geographic and health data were immediately brought together in order to optimize coordination and rescue and relief support. Hence eHealth was essential to avoid health catastrophes (epidemics) over and above the initial catastrophe[15].

EHealth is increasingly becoming an important field in the development of health care, providing real time data, and providing recommendations that are essential to help health care practitioners and institutions[16].

2.2. Clinical Decision Support Systems

Clinical Decision Support Systems (CDSSs) are information systems designed to improve clinical decision making. They are fed with the characteristics of a certain patient in order to match them to a computerized knowledge base and patient specific solutions[17]. There are several areas where CDSS are being tested.

Reminder Systems for preventions have been beneficial in 76% of the studies where performance outcomes that included rates of screening, counseling, vaccination, medication and identification of risk behaviors were measured.[18] Systems for disease management, according to 40 studies of CDSSs for active health conditions, improved practitioner performance in 62% of those studies and in 18% of the studies patients showed improvements. Systems for drug dosing and drug prescription, according to 29 studies, improved practitioner performance in 15 studies (62%) and 2 out of 29 patients showed improvement in their condition[7].

Although these systems are far from being implemented in health care systems, there is currently a widespread enthusiasm for introducing EMRs, computerized physician order entry systems, CDSSs and CPGs into hospitals and outpatient settings. In every other commercial, industrial and scientific areas, the introduction of computer systems have improved safety, readiness and efficiency. Considering this, computerization of health care should have tremendous benefits.

Combining CDSSs with CPGs can actively contribute to improvements in health care systems, having important data during clinical practice combined with CPGs can improve and remove inconsistencies from the decisions of practitioners, removing the responsibility associated from the decision by adding statistically valuable data to it. This would actively make the health care system more evidence-based which in turn will improve health care and reduce occupational stress associated with medical errors.

2.3. Clinical Practice Guidelines

Clinical practice guidelines are systematically developed statements to assist practitioner and patient decisions about appropriate healthcare for specific clinical circumstances[13].

2.3.1. Definition and Objectives

CPGs consist of a set of rules, principles or directions to assist the health care practitioner with practice care decisions about appropriate diagnosis, therapeutic, or other procedures for different clinical circumstances[13].

The purpose of developing Clinical Practice Guidelines is to summarize and catalog medical knowledge and innovations, a set of rules and descriptions which are appropriate to a specific clinical situation according to medical consensus, clinical trials and medical evidence. CPGs should also be used as a source of knowledge for new professionals and serve as guides of good practice during clinical practice to avoid medical error and standardize medical procedures[13].

2.3.2. Advantages and Limitations

There are several benefits associated with the use of CPGs, contributing with benefits to both patient and health care practitioners such as:

- CPGs can improve practitioner decisions and general practice (e.g. by simply making sure none of the stages of a specific procedure is forgotten) which consequently is an improvement for the patient that can rely on better general practice;
- CPGs facilitate the reuse of knowledge and the standardization of practice, by allowing a certain health practice to be adapted or even applied to a different procedure or guideline, which in some instance of its execution might need a specific practice which has a guideline already contributing to general efficiency and good practice;
- Guideline authors are encouraged to apply rigorous formal techniques, which helps to ensure syntactic, logical and medical correction of CPGs, making them suitable for computer research and use.

Despite these clear advantages, CPGs have yet to establish themselves as daily tools for medical practice, because of some limitations. There still isn't a simple way to use them, most guidelines are still in paper format which proves hard to use during practice, and most of the tools developed for executing CPGs on computer applications are mostly for research purposes, having difficult interfaces and lacking some essential usability required for daily use[19][20][21].

2.4. Computer-Interpretable Guidelines

As an answer for the limitations of paper CPGs the concept of CIGs appeared. CIGs are computer-based representations of CPGs that can be integrated in computer systems to support decision making and aim to increase the use of CPGs on daily basis practice.

The availability of CPGs during the clinical practice is believed to increase the acceptance and use of guidelines and general practice, by being able to observe and monitor health practitioners.

2.4.1. Obstacles

The difficulty of converting any non-computer-based format is a challenge in every computer science field and CIGs suffer from the same problem, most of the current CPGs were not designed to be digitally represented and computer-interpretable, since they imply complex instructions and the manipulation of too many variables which leads to non-deterministic and complex algorithms[7].

Another issue is the different vocabulary used by different CPGs, the use of evasive terms to quantify measures instead of numerical limits, unspecific criteria in decision making and the general lack of precision and ambiguity makes them almost impossible for computers to interpret.

The integration of an EMR is important for CIGs, but that integration at the moment is not easy to achieve due to some organizational barriers, with each organization having different data representations to store patient's information poses another challenge to develop applications for execution of CIGs considering the need to answer a huge variety of formats.

2.4.2. Requirements for the Implementation

To solve the issues with the implementation of CIGs, one of the solutions is the development of a standard model of CPG representation. A formal model that would be able to answer the complexities associated with CPGs, but also allowed them to be interpreted and executed by a computer.

A model like this could bring benefits like:

- A representation model of CPGs can be used to identify the requirements for a decision to be made, aiding professionals during the critical process of decision making in health care[22];

- CIGs add verification and validation to CPGs, by both allowing them to be automatized by computer applications and adding syntactic, logic and clinical correctness[23];
- Reutilization of knowledge is also an advantage of CIGs, by allowing other CIGs to use fragments and modules of another that along the execution of the guideline share a mutual objective, contributing for easier and faster development of new CPGs;
- The spreading of CPGs in a computer format is significantly easier and contributes to knowledge spreading and standardization of health practices.

2.5. Creating and Executing

There are four essential areas that have to be considered when developing computer applications for decision support based on CPGs:

- Modelling and representation of CPGs;
- Acquisition of CPGs;
- Verification and trial of CPGs;
- Execution.

All these aspects act as parameters when analyzing the different approaches on CIG modelling and must be analyzed in more detail [24][23].

2.5.1. Modelling and Representation

Representation of guidelines is crucial in the development of a model of CIGs, which must be able to offer comprehension of the complexity of clinical procedures present in a CPG, a precise and correct description of the guide and the ability to be parsed into machine language that allow them to be automatically executed by a computer application. Reaching these objectives requires a set of matters to be approached[25]:

- Primitives: are units of construction that are used to assemble guidelines. (e.g., tasks, rules, nodes etc.) Most of the approaches model guidelines according to the Task Network Model (TNM), modeling the hierarchy of the workflow according to a network of tasks that can acquire different forms. TNMs are normally based on a set of generic tasks such as decisions and actions;

- Complexity: the representation must be able to represent the different kinds of guidelines despite the differences between them, both in complexity and abstraction;
- Variety of knowledge: the guidelines are composed of different kinds of knowledge, namely the declarative knowledge (e.g. specific domain knowledge) and the process knowledge (e.g. inference method or the decision support method). The model must have means to express and support these types of knowledge to allow CIGs to be used in different clinical fields;
- Language: the representation must support a formal language (vocabulary, syntax and semantics). The language must be compact and objective to allow real time automatized execution of guidelines during the clinical process;
- Adaptation: it is important that the model is generic to allow the sharing of CIGs throughout different institutions. To achieve this, there is the need to have mechanisms to update guidelines and spread the changes to the specific adaptations on each institute.

2.5.2. Acquisition

Medical guidelines are evidence-based statements resulting from clinical trials and existing literature, the processes must be based on the adapted model to represent CPGs and should be aided by tools to help the specialists to create and structure the data according to the model. Mechanisms to update are necessary to the creation of new versions of the guidelines

2.5.3. Verification

Verification consists of two aspects: if the medical guideline is well made or which of two different guidelines is the best. The first question seeks to find the formal correctness of the guidelines. The second one is a bit more difficult to answer because there is the need to evaluate which one produces the best outcomes[26].

2.5.4. Execution Aspects

As mentioned above, a medical guideline contains great amounts of information to be considered such as decisions, constraints and temporal restrictions. These are all aspects to be considered during the execution, the system should be able to collect them during the execution of the guideline[26].

Ideally a guideline execution engine should meet the following requirements:

- Keep a repository of guidelines;
- Facilitate the creation and maintenance of guidelines through an easy to use interface;
- Provide a formal language to represent CPGs;
- Provide mechanisms to coordinate the services required during the execution of guidelines;
- Allow the analysis of the behavior of the guideline through the automated execution;
- Provide a connection to a computerized patient record;
- Allow the use of standard terminology inside guidelines;

The architecture of the execution engine should not depend on the system and the application used, because it would not allow the use of the engine in different clinical domains.

2.6. Research Projects and Applications

Here the different projects and applications regarding CPG automated executions are summarized. There are some projects that are worth being mentioned, according to Isern and Moreno's review of the current CPG execution solutions[26], the engines with most impact currently are:

- Arezzo
- Digital Electronic Guideline Library (DeGeL)
- Guideline Acquisition, Representation and Execution (GLARE)
- GLIF3 Guideline Execution Engine (GLEE)
- Health Care Services (HeCaSe2)
- NewGuide
- Standards-based Sharable Active Guideline Environment (SAGE)
- Specification Execution and Management Plan (SpEM)

When executing CPGs the main focus is the analysis of the representation and modelling of CIGs as well as execution aspects. Next there is a review and analysis of these execution engines.

2.6.1. Arezzo

The Arezzo® clinical decision support technology enables the design, creation, and execution of clinical pathways, guidelines and patient care protocols that present patients and medical professionals with evidence-based advice for each individual patient[27].

- ***Clinical Guideline Representation***

ArezzoTM uses the PROforma language to represent CPGs. PROforma is an executable process modeling language that has been successfully used to build and deploy a range of decision-support systems, guidelines and other clinical applications. It has a declarative format defining four basic types of tasks (plans, decisions, actions and enquiries) as well as logical and temporal relationships between them. An action is a procedure to be carried out (usually by an external element like a doctor or a medical resource). A plan is the basic building block of a clinical guideline and represents a container for a number of tasks, including other plans. A decision is a task that represents an option in terms of different logic commitments to be accomplished. An enquiry is a request for further information or data required before proceeding with the application of the guideline[28].

- ***Architecture***

The tool is composed of three elements: a Composer, a Tester and a Performer (see Figure 1). The Composer is used to create guidelines using the PROforma language. The Tester is used to test the guideline logic before deployment (it checks that the statements in decisions, tasks and inquiries are well written). The Performer inference engine can then run the guideline, taking into account data related to patients stored in existing health care systems EMR[27]. During the enactment of a guideline in the performer engine, a task changes its internal state depending on whether the task is waiting for data, suspended, finished, or it cannot be accomplished in the current state of the patient. The instance of execution is linked to the electronic medical record (EMR) by personalizing the execution.

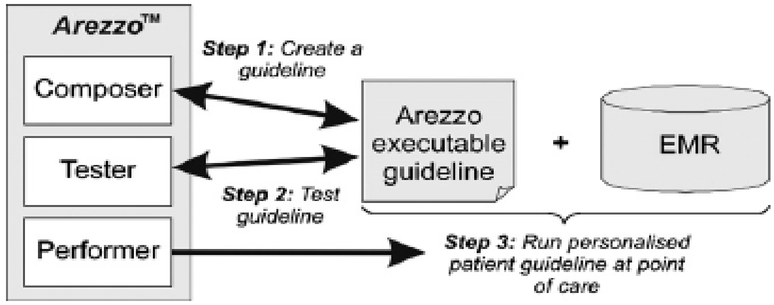

Figure 1 - Arezzo architecture (extracted from [27]).

2.6.2. DeGeL

Digital Electronic Guidelines Library (DeGeL) is a web-based modular and distributed architecture, which facilitates gradual conversion of clinical guidelines from text to a formal representation in Asbru[29][30]. It is being developed in Ben Gurion University in Israel.

- ***Clinical Guideline Representation***

The system maintains a repository of guidelines, and it allows the user to search, browse, retrieve and visualize all available guidelines. At the moment, the system creates guidelines using the formal language Asbru[31], but the methodology could be extended to other languages. One of the goals of DeGeL is to create formal guidelines from textual documents. The initial textual guidelines go through an intermediate layer between the textual and the final form, where experts add semantic information. The intermediate layer uses a meta-ontology that defines a hierarchy of basic concepts.

- ***Architecture***

DeGeL is a modular system composed of a set of tools that support guideline classification, semantic mark-up, content-sensitive search, browsing, run-time application, and retrospective quality assessment (Figure 2).

It's composed by a collection of different tools. *Uruz* is a tool that allows medical professionals to create guidelines, *IndexiGuide* another tool this time to retrieve guidelines. The execution engine is composed of a set of tools to test and visualize CPGs such as *VisiGuide* to visualize and browse CPGs; another tool is *Vaidurya* which allows both searching and retrieving CPGs. Finally *Dipole* is a tool to assist practitioners to determine patient eligibility and guideline applicability. It uses different standard to define medical vocabulary as well as a patient database to store the information of the users.

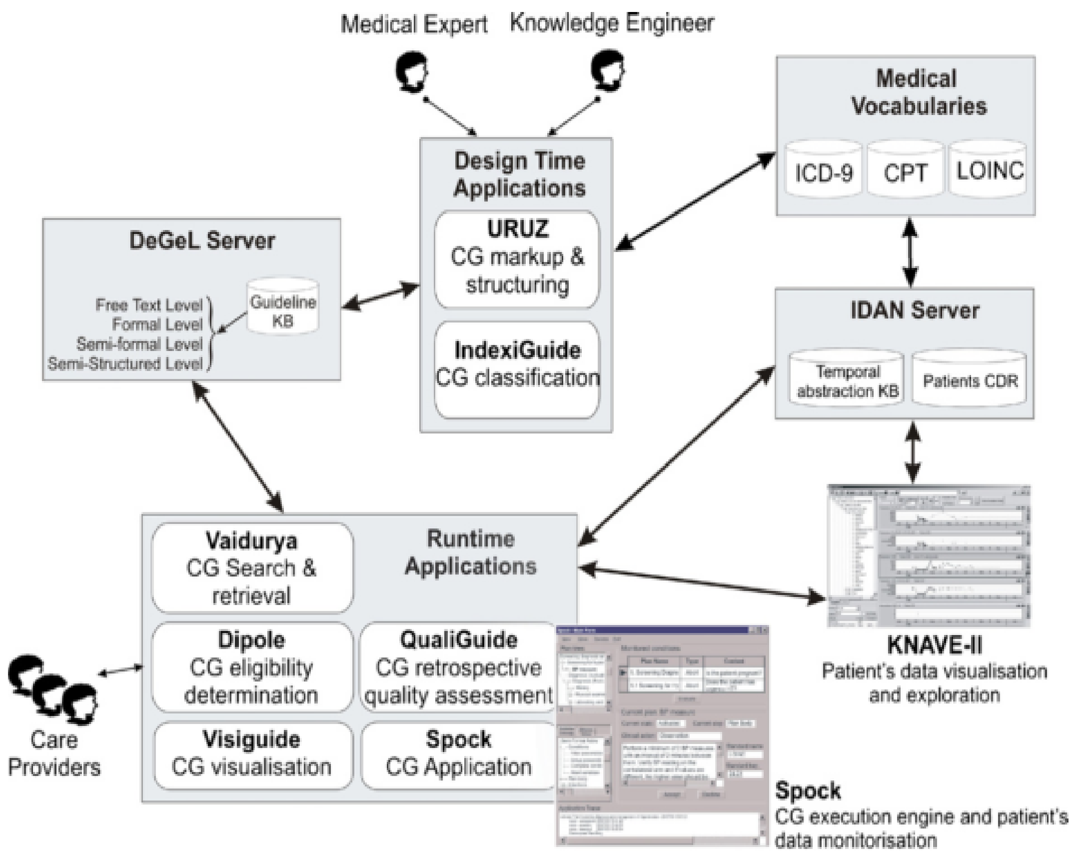

Figure 2 - DeGeL general architecture (extracted from [30]).

2.6.3. GLARE

GuideLine Acquisition, Representation and Execution (GLARE) is a system to acquire and execute CPGs, it was developed at the Computer Science Department at the Università del Piemont Orientale of Alessandria (Italy) with cooperation of an Hospital in Italy (Azienda Ospedelieria San Giovanni Battista of Torino).

- ***Clinical Guideline Representation***

GLARE doesn't use any of the best-known representations, the developers defined a proprietary graph-based representation, where each action is represented by a node, while control relationships are represented by arcs[32]. The model distinguishes two different kinds of actions, atomic and composite as we can see in Figure **3**. Atomic as the name suggests, are simple actions to be executed during any point of the CPG. There are four possible atomic actions defined:

- Queries, that allow the request of any external information;
- Work actions, that represent actions to be performed;
- Decision actions, a set of conditions to select an alternative among a set of actions that could be performed at that instant.
- Conclusions that allow describing outputs of a decision primitive.

Composite actions are a collection of atomic or other composite actions. For each action there is a set of preconditions, to be fulfilled before its activation, and a set of conclusions, that hold after the execution of the action.

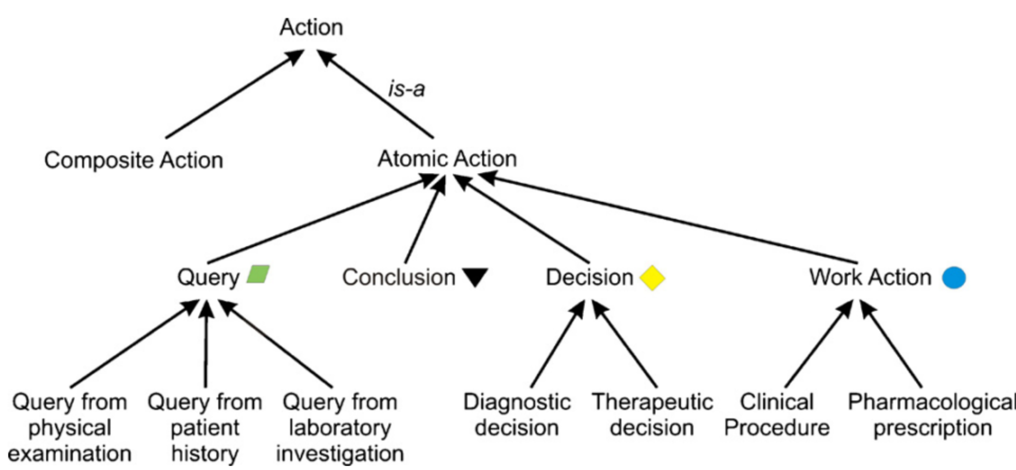

Figure 3 - Basic entities defined on GLARE's Representation model[32] (extracted from[33]).

- ***Architecture***

GLARE's architecture distinguishes two different phases, the first being the CPG acquisition, i.e. when a guideline is introduced in the system, and the execution phase, i.e. when the CPG is applied by a practitioner.

There are three layers (see Figure 4): System, eXtensible Markup Language (XML) and System layer. The bottom layer makes the connection between the database and upper layer to get access to the stored data such has the patient database, CPG database and so on. The middle layer structures the data from the databases into XML to be parsed by the upper layer. The system layer manages time constraints between different actions on the CPG and the Execution module allows the execution and simulation of a CPG using the retrieved data from the databases according to the specific situation.

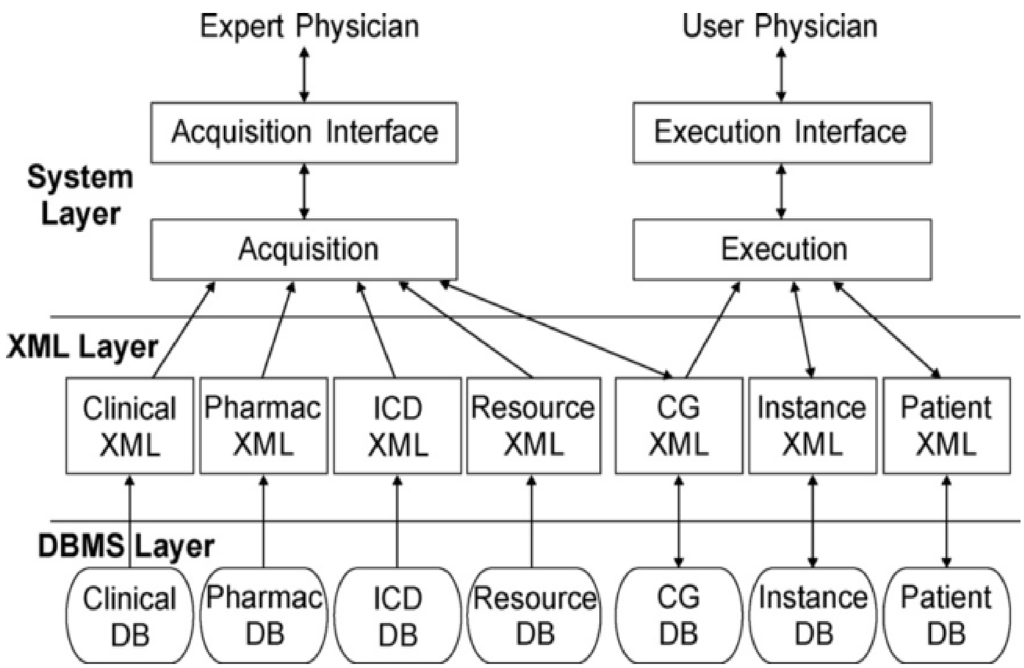

Figure 4 - GLARE General Architecture[32] (extracted from [33]).

2.6.4. GLEE

GLEE is a tool to execute CPGs encoded in GLIF3, the third version of the GLIF standard, developed across different institutions in the United States [34][35].

- ***Clinical Guideline Representation***

The representation used by GLEE for CPGs is GLIF3. GLIF3 represents guidelines as flowcharts of temporally ordered nodes called guideline steps that store actions (Action Steps), decisions (Decision

Steps), and clinical states of the patient (Patient Clinical States). There are two more types of nodes, called Branch Steps and Synchronization Steps, which are used for modeling multiple concurrent paths through the guideline[36].Decision criteria are modeled using an OCL-based language (Object Constraint Language) called GELLO [34].

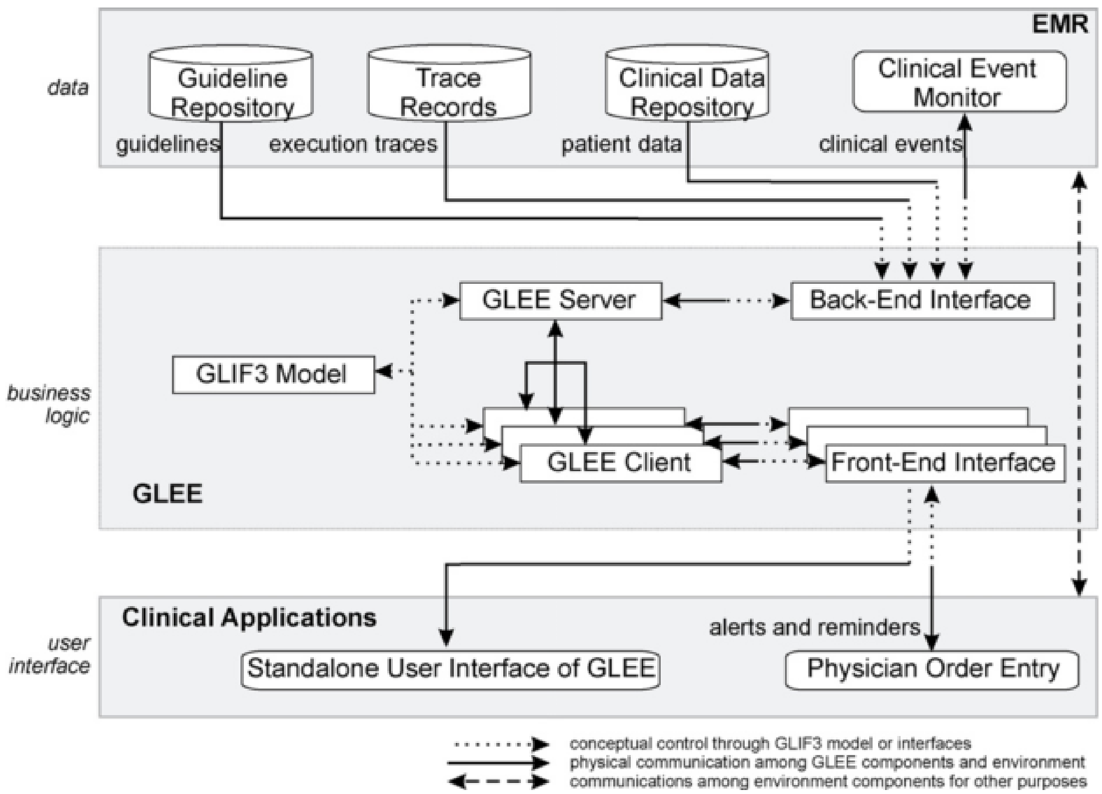

Figure 5 - GLEE general architecture (extracted from [37]).

- ***Architecture***

Then again like GLARE, the architecture is divided into three different layers as seen in Figure **5**, acting with similar purposes, but taking a bit different approach as we will see. The data layers contain the CPGs and the EMR with a CPG repository and the clinical event monitor that allows the execution and simulation of CPGs through an event driven model. The business logic serves as an intermediary between the user interface layer and the data layer, by serving as a server for the requests made by both the other layers.

The execution model of GLEE takes the *system suggests-user controls* approach. A tracing system is used to record an individual patient's state when a guideline is being applied to that patient. It can also support an event driven execution model once it is linked to the clinical event monitor in a local environment. The tracing system allows maintaining two main views of the execution. On the one hand, GLEE suggests which actions can be performed and decides which actions (whose preconditions are satisfied) can change the state from started to finished. On the other hand, the user can control the process, and it can initiate, confirm or decide different transitions between actions. Figure **6** shows the whole process maintained by the GLEE execution engine through the tracing system.

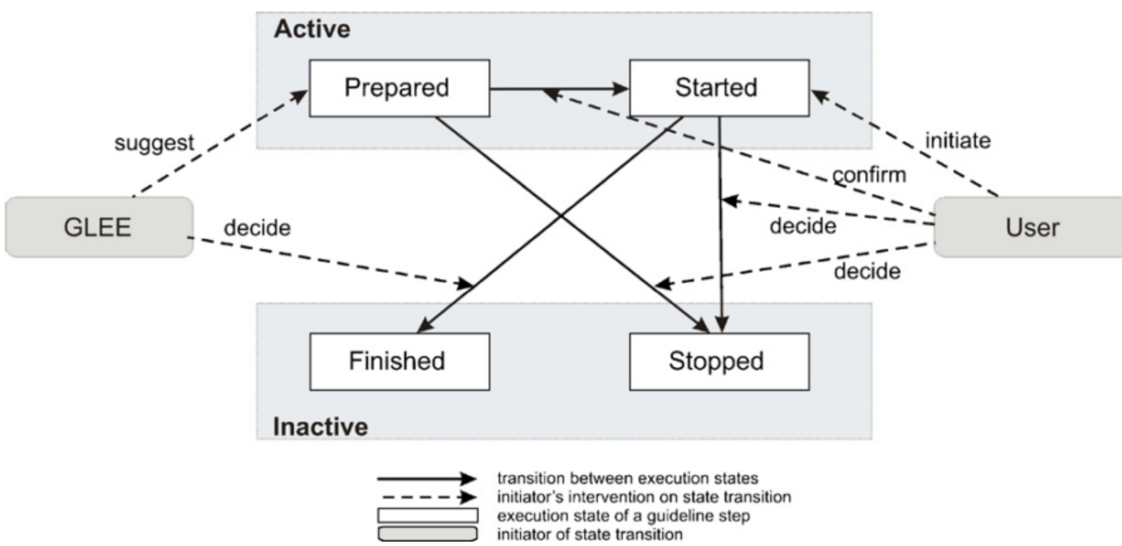

Figure 6 - GLEE state transition model[35] (extracted from [38]).

2.6.5. HeCaSe2

Health Care Services Release 2 is an agent-based[39] platform that offers health care services to users (patients and practitioners) developed as an academic prototype at the Intelligent Technologies for Advanced Knowledge Acquisition (iTAKA) at the Universitat Rovira i Virgili, Catalonia (Spain). The platform uses agents with different roles with multiple interactions between them, humans and medical devices.

- ***Clinical Guideline Representation***

As its CPG representation language, HeCaSe2 uses PROforma[28]. In addition to PROforma, it uses the Unified Medical Language System (UMLS) to enrich its queries and actions with medical terminology. UMLS[40] defines a Code Unique Identifier (CUI) that is used to identify all medical terms.

Internally, the structures of the medical organization as well as the medical terms are stored on a single ontology. The medical ontology provides a common semantic framework to execute CPGs, and the agents can coordinate their activities with other agents or entities.

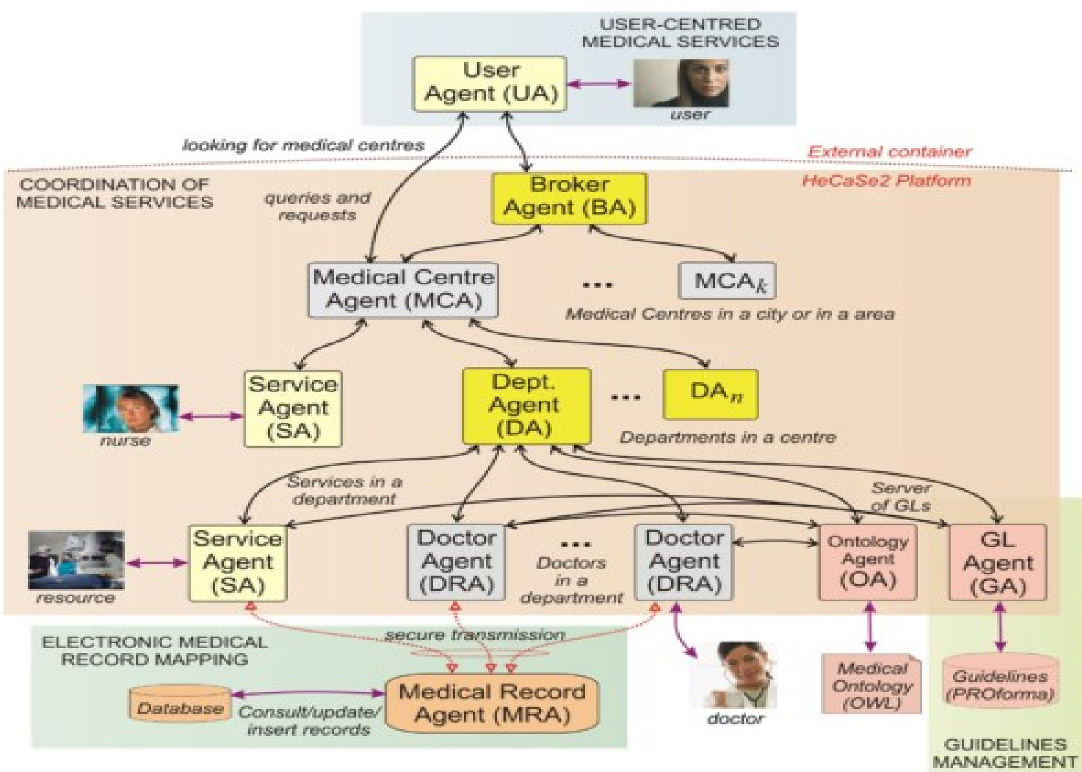

Figure 7 - HeCaSe2 general architecture (extracted from [41]).

- ***Architecture***

The authors propose an architecture consisting in various kinds of agents interconnected and acting independently with their own knowledge and data. The basic architecture being developed by this project can be seen on Figure 7. At the top there is the patient who interacts through a user agent storing information and data related to the user. This agent interacts with a broker agent with the information with all the medical centers in the area. The MCA agent manages and monitors all of its departments and resources. Each department then has his own agent responsible for the doctor agents which are aware of the services that they can provide. The Guideline agent is responsible for everything related to CPG execution. Lastly, the Medical Record Agent controls the access of the database that stores all the medical records related to the patients. Using the CPG, a doctor can consider all the available data and make an informed decision. During the visit, if the doctor needs another test to be performed on the patient, agents negotiate the best alternative according to the preferences/constraints of the user, the doctor and the hospital services[41]. The medical services needed in the execution of a guideline can be located in the same medical center or in another one (found with a previous discovery process).

2.6.6. SAGE

Standards-based Sharable Active Guidelines (SAGE) is collaboration among researchers from the United States. The project aims at two things, to create an infrastructure for clinical experts to author and encode CPGs using standard representation and then to use this infrastructure to deploy these CPGs across heterogeneous clinical information systems.

- ***Clinical Guideline Representation***

SAGE uses the EON formalism, which is a set of Protégé classes and plugins[42]. Sage defines two different formalisms, recommendation-set is an activity graph composed process and interactions between them, the graphs allow the specification of algorithms and medical planes. It uses context nodes such as patient information, decision nodes that consider specific condition variables and action nodes which determine a set of tasks to be performed by either a practitioner or a computer.

- ***Architecture***

The central component of the SAGE architecture is the guideline execution engine as it can be seen in Figure 8. When given a guideline, the engine collects the required data from an internal repository and allows medical experts to emulate real guideline behavior[26]. It uses standardized components that allow interoperability of guideline execution elements with the standard services provided within vendor clinical information systems. It proposes the use of a repository of CPGs in order to manage all available guidelines. The engine includes organizational knowledge to capture workflow information and the resources needed to provide decision support in enterprise settings. It proposes a methodology to develop/create medical guidelines[26].

It offers integration with a standard EMR which connect with the execution engine to store the information provided by the execution engine.

It includes organizational knowledge to capture workflow information and the resources needed to provide decision support in enterprise settings.

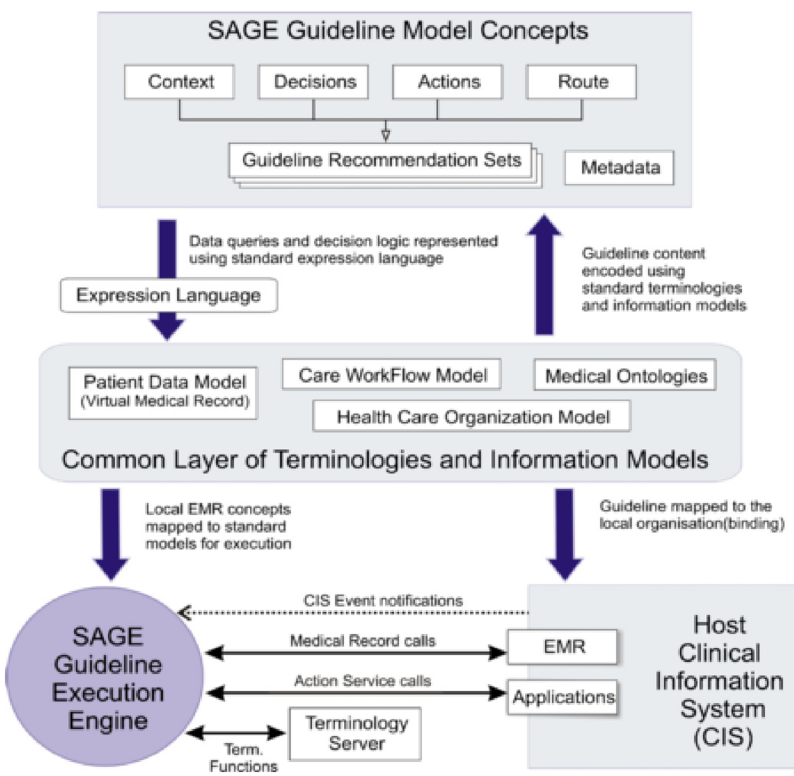

Figure 8 - SAGE general architecture (extracted from [42]).

2.7. CompGuide

CompGuide is a PhD project currently being developed at the Intelligent Systems Lab (ISLab) , in the University of Minho with the main objective to provide health care professionals with formal models and tools for construction, maintenance and executions of CPGs, i.e. using software applications for its analysis, development and deployment. The project presents itself with the following objectives:

- A prescriptive model for integration and iterative development and analysis of CPGs;
- Integration of the guideline model with standard clinical terminologies and data models
- A set of software tools to support such a development and analysis process;
- A library of guideline items with re-usable knowledge patterns;
- A clinical decision model that combines guideline recommendations with techniques that provide contextual information about the environment (e.g., place, season of the year) and deal with incomplete information;
- Deployment of several case studies using a prototype to acquire and execute guidelines.

A prototype that implements the decision model will allow the development of case studies using CIG versions of guidelines that are being used in health care institutions and known clinical cases provided by The School of Health Sciences at the University of Minho[43].

And contrary to the other projects and applications described above, aims to gather together in a pack, different applications and technologies that are inter-connected to create a multi-objective platform. We will focus more on this project considering this thesis as a component of CompGuide.

2.7.1. CompGuide Representation Model

Contrary to the other projects, CompGuide takes a different approach by conceiving a new model to store and represent CPGs using Web Ontology Language (OWL). OWL is a standard developed by the World Wide Web Consortium (W3C) designed to be used by applications that require the processing of information rather than just presenting information to humans.

The model was developed to fulfill all the requirements of CPG execution, i.e., representation of administrative information, construction of workflow procedures, and the definition of temporal and clinical constraints at once by not having to rely on different tools and applications like the other projects studied above.

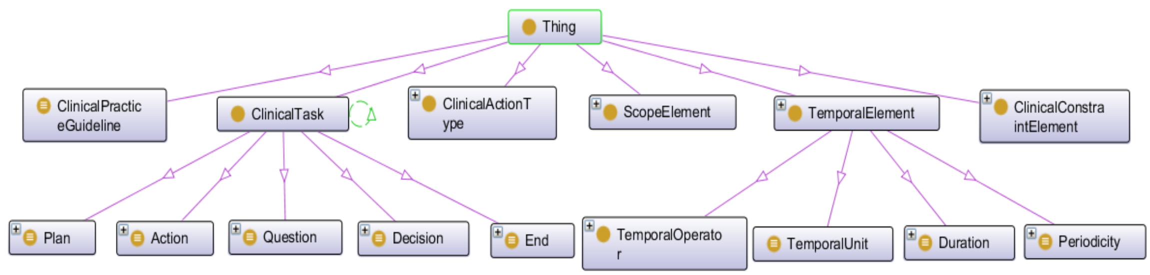

Figure 9 - Main primitive classes in the CompGuide model.

Looking at Figure 9, we can see that a CPG is an instance of a *ClinicalPracticeGuideline* class, the individuals of this class contain a set of data properties to allow versioning and rigorously describe the objectives and content of the CPG. OWL has built-in data types that allow the expression of simple text, numeric values and dates. The data properties defined for administrative purposes were *Authorship, guidelineName, guidelineDescription, DateOfCreation, DateOfLastUpdate, and Version Number*. Other optional properties were defined to specify in which conditions a CPG is applied such as *ClinicalSpecialty, GuidelineCategory, intendedUsers, and targetPopulation.*[44]

2.7.2. Construction of Workflow Procedures

CPGs are composed of a series of tasks that may or may not have to be performed during a clinical encounter. To emulate these tasks, they are defined under the *ClinicalTask* class, through the subclasses *Question*, Action, *Plan* and *Decision*. The tasks of an instance of *ClinicalPracticeGuideline* are all contained in an individual of *Plan*, to which is linked through the *hasPlan* object property. The *Plan* class is an important class that will allow the reuse of information. A *Plan* is then connected to its tasks through the *hasFirstTask* property which then can be linked to another task by the *nextTask* property. The *Action* class represents a step that has to be performed by the practitioner that includes clinical procedures, clinical exams, and medication or non-medication recommendations.

The *Decision* class serves the purpose of representing choice, by allowing the decision between two or more options, this decision can be associated with rules and options through the use of object properties that connect them to *ClinicalContraintElement*. Normally it is used to infer something about the state of a patient, namely in diagnostic situations. To assure that only one path is chosen, the class *alternativeTask* was implemented to represent the choice of paths. CompGuide also offers *parallelTasks* to allow cases that require two of more tasks to be performed simultaneously.

The purpose of the *Question* class is to interact with the practitioner about a patient's information, being used to obtain the required information necessary to continue the CPG execution. There are data properties created to specify the name of the parameter to be obtained, the values it may assume and the units in which it should be expressed.

The End class is used to signal the termination of the execution thread. A CPG extract from the Standards of Medical Care in Diabetes guideline from the American Diabetes Association expressed in this model can be found in Figure 10 [44].

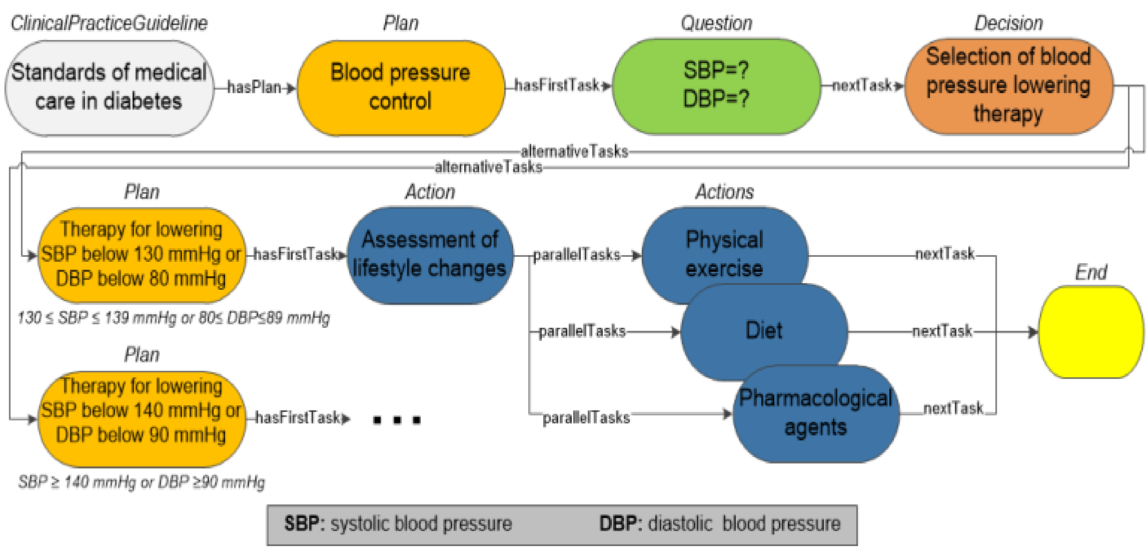

Figure 10 - Excerpt of the Standards of Medical Care in Diabetes clinical guideline.

2.7.3. Temporal Constraints

Time constitutes an important factor when performing any clinical procedure, the time at which the patient is showing the symptoms and uses that information to improve the solution obtained. It's important then the ability to represent time when executing CPGs, to do that CompGuide defines *TemporalElement*. This class contains two constructors, *Duration* and *Loop. Duration* has a double data type property called *DurationValue*, where a value for the intended duration of either *Actions* or

Plans is provided. The *TemporalUnit* class, also defined under *TemporalElement*, covers individuals that represent the different time units in which the *Duration* may be expressed, namely *second*, *minute*, *hour*, *day*, week, *month* and *year*. In *Periodicity* cycles can be represented, by each instance of *Periodicity* having a *repetitionValue* where the number of repetitions of the cycle is set. Each instance also allows the definition of an object property *hasTemporalUnit* and *periodicityValue* data property that can be used to set the interval between task executions. The Periodicity class also encompasses another situation which is the definition of stop conditions for cycles. These conditions correspond to updates in the patient sate that cause the interruption of a cyclical task. The object property *hasStopCondition* enables the definition of these particular cases.

2.7.4. Clinical Constraints

As mentioned before, a decision implies a choice which requires constraints to be achieve, these constraints can be set by the use of the *hasOption* property that connects to individuals of *Option*. Each *Option* has a *Parameter* and a *NumericalValue* or *QualitativeValue* data property. The rules that dictate the option selection are provided by the *hasConditionSet* property, linking the individuals from this task to *ConditionSet*. The last one gathers all the necessary conditions through *hasCondition*. In Condition, it is possible to define the clinical parameter whose value will be compared, the unit it should be in and the operator that should be used. The *hasComparisonOperator* property connects individuals from Condition to the *ComparisonOperator*[44]. The set of comparison operators defined in this class include: different from, equal to, greater or equal than, greater than, *TriggerCondition,* which also uses *ConditionSect,* signals the choice made by the *Decision* class.

2.8. CompGuide and the Other Models

CompGuide offers a different approach from the alternatives above, by using an ontology to represent CPGs. Ontologies offer a tool that can be used to more easily represent information that can be processed, and this is where CompGuide presents a solution that makes use of those features, offering the definition of clinical constraints, temporal properties, clinical task scheduling and how all these aspects connect with each other. CompGuide provides more expressive power by allowing the definition of a clinical workflow, similarly to GLIF3 and PROforma. However, these models do not have native methods for expressing temporal constraints, using a subset of Asbru temporal language to deal with this issue. CompGuide also offers a significantly more complex temporal constructs and allows an easy reuse of information.

CompGuide was clearly inspired by some of the projects and applications studied, it offers the best functions of each, by compiling those features into a usable application using updated technology such as Java, Web Services, OWL, JSP and possibly Android allowing the application to be multi-platform and easily expandable.

3. Development of a Computer Interpretable Guideline System

3.1. Technologies and tools

When developing new applications in the computer science field, a wide number of different technologies, tools, and frameworks are available to use, the choices that were eventually made, will be lightly covered on this section. Some of those choices were made considering the requirements of the CompGuide project's architecture, while others that were not so dependent on the architecture, but reflected both research and personal preference. The usage of web services offer expandability and improvement of different servers without compromising the other services provided, offering easily expandability of the application to add features when required.

3.1.1. Web services

Web services are a form of communication between two electronic devices over the World Wide Web; they offer a simple way to develop Web and Mobile applications, by separating the processing required such as database manipulation and heavy data computation to a specific server who hosts the web services, allowing for smaller and faster applications[45].

The project uses Representational State Transfer (REST) web services; they offer a simple way to use them by using native HTTP operation and features such as GET, POST, UPDATE and DELETE[46].

So, developing the application as a Web Application Programming Interface (Web API) offered everything the project required, such as multi-platform capabilities, scalability and the ability to extend the application without the necessity to have to change what is already implemented[47].

3.1.2. Web Ontology Language (OWL/OWL-API)

The Web Ontology Language (OWL) is a family of knowledge representation languages for authoring ontologies. The languages are characterized by formal semantics and RDF/XML-based serializations for the Semantic Web. OWL is endorsed by the W3C and has attracted academic, medical and commercial interest. The use of the OWL's XML presentation syntax to obtain Web services that provides reasoning support and easily deal with facts and rules [48].

The advantages of OWL reside in the manner a system uses the information. Machines do not yet grasp human language and, occasionally, there is content that escapes their understanding. For instance, a human being may comprehend that in some situations there are words that are unquestionably related, although not being their replacements. A machine does not recognize these relationships, but semantics are essential. Indeed, the idea behind OWL is to provide a machine with a semantic context; the advantage relies on the creation of a better management of the information and its descriptions.

Integration with Java is also easy to achieve by using OWL-API, an open source Java API and reference implementation for creating, manipulating and serializing OWL Ontologies developed and maintained mainly by the University of Manchester[49].

Given the fact that CompGuide's[44] repository is stored in an OWL file, it's required to use OWL and OWL-API in order to integrate with the rest of the project.

3.1.3. Java

Java is currently one of the most used languages in computer science, by using a java virtual machine, Java applications only need to be coded once and run on a virtual machine, so it can be executed in any computer without the need to recompile or recode anything as long as that machine can run Java virtual machine.

The spreading of the language made it one of the most complete, having APIs for virtually everything and being one of the few with an OWL API. The features available make Java a suitable option for the needs of this project such as Java Server Pages and JavaBeans Open Source Software Application Server (JBoss) [50].

- **JavaBeans Open Source Software Application Server (JBoss)**

JBoss is a simple and yet powerful application server, which fulfilled the requirements for the implementation of the application, it offers powerful tools such as web service publication, filters, and even authentication backed up by JAVA, allowing almost anything to be implemented due to java's enormous amount of libraries.

- **JavaServer Pages**

JavaServer Pages is a technology that allows the creation of dynamically generated web pages, although the application could have been implemented using any other available technology, it seemed a good option to choose java as it allowed the re-use of segments of code, consequently reducing the work required and still being very powerful and deployable with the Jboss application server, so, for testing, this was convenient.

3.1.4. JSON/Gson library

JavaScript Object Notation (JSON) is a text-based open standard designed for human-readable data interchange. Derived from the JavaScript scripting language, JSON is a language for representing simple data structures and associative arrays, called objects. Despite its relationship to JavaScript, JSON is language-independent, with parsers available for many languages[51].

In the context of this application JSON is used as the format in which the data is transferred between the server and the web application. The Gson[52] library is used to do the serialization and deserialization of the transferred data for the correspondent classes. It offers simpler parsing and it is compact when transferring small chunks of data from server to client, making it a better alternative than the more standardized XML by simply making it easier and lighter which is one of the goal of the application

3.1.5. MySql

MySql is a Relational Database Management System (RDMS) widely used for web applications, it is the most used open source RDMS, it offers simple tools like MySql workbench which makes developing the model easy. It is also very easly integrated with both JBoss and Java. It's widely used and well tested, and offers many tools to create/edit database relational models. It became also an easy choice due to multi-platform compatibility.

3.1.6. Protégé

The Protégé-OWL editor is an extension of Protégé that supports the Web Ontology Language (OWL). OWL is the most recent development in standard ontology languages, endorsed by the World Wide Web Consortium (W3C) to promote the *Semantic Web* vision. An OWL ontology may include descriptions of classes, properties and their instances.[44]

Protégé-owl was mainly user to get a visual representation of the guideline model, to more easily develop the OWL handler and parser.

3.1.7. Twitter Bootstrap

Bootstrap is a free collection of tools for creating websites and web applications. It contains HTML and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions.[53]

It was used to simplify the development of the web application visual, to make it simpler and easier to use, while making it look presentable.

3.2. System Architecture

The developed system is composed of four main components, each responsible for specific tasks, given Comp Guide's model, this system maintains the idea of separated components, to allow an easy deployment, customization and even extension of the system. The system makes use of web services to communicate between the main applications, such as the web application and in the future, a possible mobile application.

Figure 11 shows, in a simple way, the developed architecture, the data storage components are our own database model to store user, patient and execution information and the guideline repository which stores the guidelines present on the system.

The server acts as an intermediary between the data storage and the user interface, it implements the connections and the guideline execution which is fed through web services to the user interface.

3.3. Core Server

The core server consists in an application that makes use of JBoss to provide all the required services to allow external applications like the Web application or a mobile application to be able to execute guidelines, serves as the main component which is responsible for all the database manipulation, guideline creation, controlling the execution, storing patient information and user authentication and all the guideline execution information that can be introduced by the user, or generated by the application.

It was developed using java and taking advantage of some of the features provided by JBoss, which allowed a simpler creation of session controllers, user authentication and publication of web services.

Within the core server, different components had to be developed in order to achieve what was proposed, these components were: the guideline handler, database manipulation, and user authentication and execution engine. These components work under the layers of web services that offer the features to the other apps, and seamless for the user interface applications.

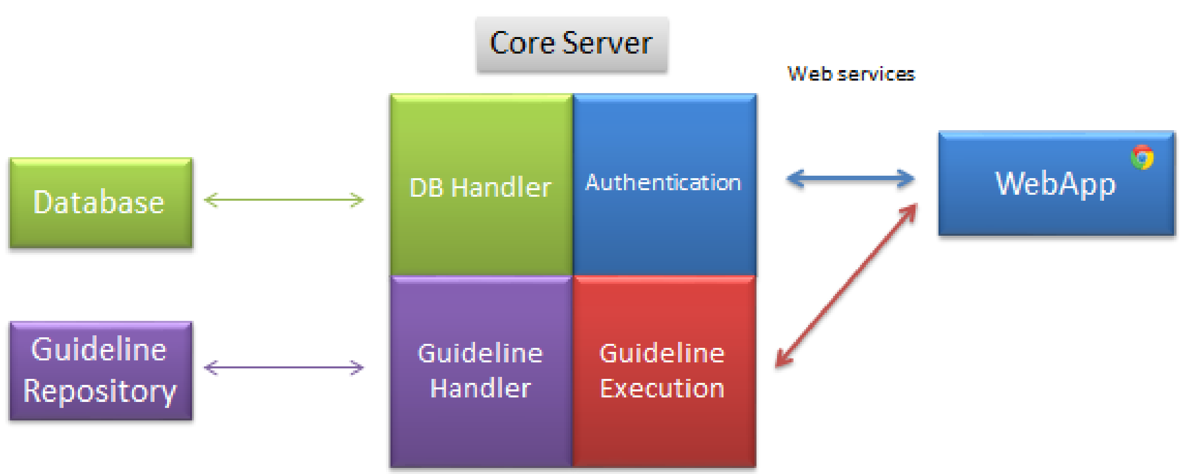

Figure 11 - CompGuide application architecture

3.3.1. Authentication and Authorization

The core server provides both authentication and authorization when accessing the services provided by the application, it allows a user to authenticate and be able to use the features that he should have access to. The system has defined two kinds of users, *admin* and *user*, *users* can only execute guidelines while *admins* can manipulate user information, patient information and guideline creation besides executing guidelines.

The process of authentication is made using a specific web service, which when provided with a user and password, it sends a response with a token generated which is stored on the database that can be used to use the request the services provided while that token does notexpire, when it does, it is required to authenticate again and generate a new token.

Authentication is done using a JBoss feature, called filters, which allows a series of checks to be done when a web service is called, making sure that the specific token sent by the user matches the token generated and stored when the authentication took place.

It's important that the information cannot be seen by anyone without access given the privacy of the information; this token assures the server that the user has indeed access and the level of access. This token is essential for all the web services to respond with the requested data.

3.3.2. Guideline handler

In order to execute guidelines, it was necessary to retrieve them from the file where they were stored, all the current case studies stored in the guideline repository, were defined by using *Protégé*, an application which serves as an OWL editor, so in order for the server to reach that information stored on the file, it was developed a series of functions to fetch the information for each different class defined on the guideline model.

While developing this parser/handler it came evident that in bigger hand coded guidelines errors could easily occur, such as forgetting specific components and properties, so a syntactic checker tool was also created, to make sure the required components are in fact defined, this guarantees that the application won't be missing necessary information and possibly lead to errors.

This component also offers methods to create and manipulate individuals of each class, which will be used by the core server guideline creation web services.

This was one of the most time consuming components to create, it required a great amount of coding and data treatment, forcing the creation of fitting classes to match the ones declared on the ontology, and the verification of the type of the data. It is prepared to process every possible type of representation of a guideline, and given the web architecture it is more efficient if the fetching of data is done only when required instead of keeping a full representation of a guideline on the server.

3.3.3. Guideline execution

The execution of guidelines is done entirely by the core-server, it is responsible for the control of the execution, task selection, patient state manipulation, collection and storing all the information provided by the external applications.

It offers two different services, which are both responsible to provide the external applications with the next task to be executed as well as all the information composing that task. The difference though, is that one stores the patient info, and considers the task executed, while the other only skips to the next task.

Whenever the web service for executing or passing to the next time is called, it proceeds to make several tasks, such as determining the next task, checking if it is a valid task, and when it finally executes it, it stores the state of execution in the database so that when the web service is next request it can re-construct the state of the execution to how it was before.

The execution web service is responsible for providing the web application with all the data required for the execution of the next task/tasks as well as the pending tasks still to be executed, when requested with the identifier of a generated task

3.3.4. Database manipulation

To access and store information on the database, the core server has implemented all the required updates, inserts and selects required by the core server to manipulate all the data which is stored, such as authentication, user information, patient information, and application generated data.

The core server offers all the required web services for the external application to manipulate the database, this offers the ability to use the authentication and authorization developed, to control the user access to the database, also maintains the database protected from external access, being only accessed by the core server.

Another time consuming task, given the complexity of the database, was the development of the required methods to access and process data from the database.

3.4. Database

Creating a database model to store all the information that can be generated during the execution of guidelines can be quite challenging, the model implemented, not only stores that information, but all the patient and user information, storing it combined with every instance of a guideline execution, allowing to map out the record of a user, patient and every instance of a guideline execution.

The different tables created to store all the information that were defined are the following:

- autenticacao – Stores the authentication tokens generated;
- patient – Stores the patient information;
- user – Stores the user information;
- guideline – Stores the guidelines currently active in the system;
- generatedTask – Stores the tasks created by the server to be executed;
- guideexec – Stores the information of an instance of a guideline execution;
- task – Stores the data of an executed task;
- exam – Stores the information of an executed exam;
- medication – Stores the information of an executed medication;
- nonmedication – Stores the information of an executed non-medication;
- observation – Stores generic information about the user state;
- procedure – Stores the information of an executed procedure;
- conditionSet – Stores the information of relevant condition sets executed.

The choice of storing all this information, is to allow the application to store as much information as possible for future executions, also creating a database of information about a patient, which can later be user to add new features such as statistics, decision making, suggestions and predicting output of a specific guideline, by comparing to older executions.

The detailed model can be seen on figure 13.

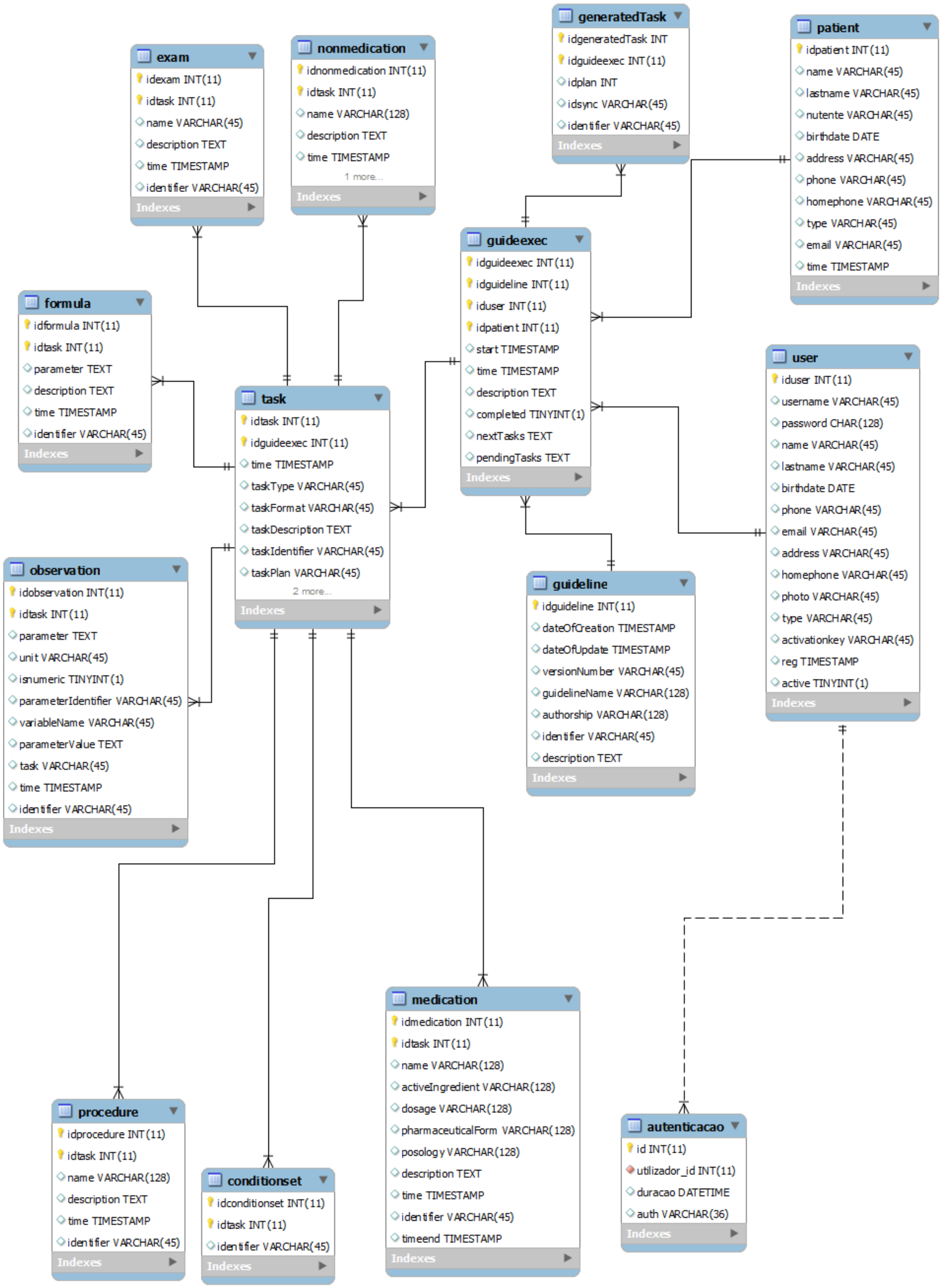

Figure 12 - CompGuide Database Model.

The method of storing the execution data by saving the important information as observation (relevant states of the patient) and actions (exam, medication, nonmedication, procedure) which store a variety of information about a patient was inspired by the model suggested by Gutiérrez[54] the model itself is design to store artifacts of a guideline execution, in a relevant structure to be later analyzed.

The expressivity and simplicity allows a good way to specify clinical concepts and storing it in an organized way that can support a wide variety of situations which present themselves in a medical environment.

3.5. Guideline Repository

The guidelines are stored using OWL, which means it's stored in a file present and accessible to the core server, contains all the different individuals that define a specific guideline, and the guidelines themselves. It's defined using the CompGuide model, and it's integrated on the system, which features an option to reload the guideline and update the database indexes of the guidelines currently in the system.

This file can be edited and updated anytime, since the server stores the information from the file into the memory for performance issues, and to allow edition without having to stop the execution, a web service is provided to update the guideline information on the server, by reading the file and reallocating the new file in the memory this action can only be performed by an admin.

3.6. Web Application

The application was developed to show and make use of the features implemented on the core server, it is responsible for the interface with the user collecting the information from the user and send it to the server. This information is then processed by the server which replies with the necessary information to allow the web application to show what comes next.

The web application features already some of the main components expected from a web application such as, it provide the user the ability to login to the server allowing the use of the application, guideline execution, and user control panel.

3.6.1. Control Panel

The application provides the users with a control panel where they can edit and manage their personal information such as name, telephone, email etc. The usage of these features is shown on **Figure 13**.

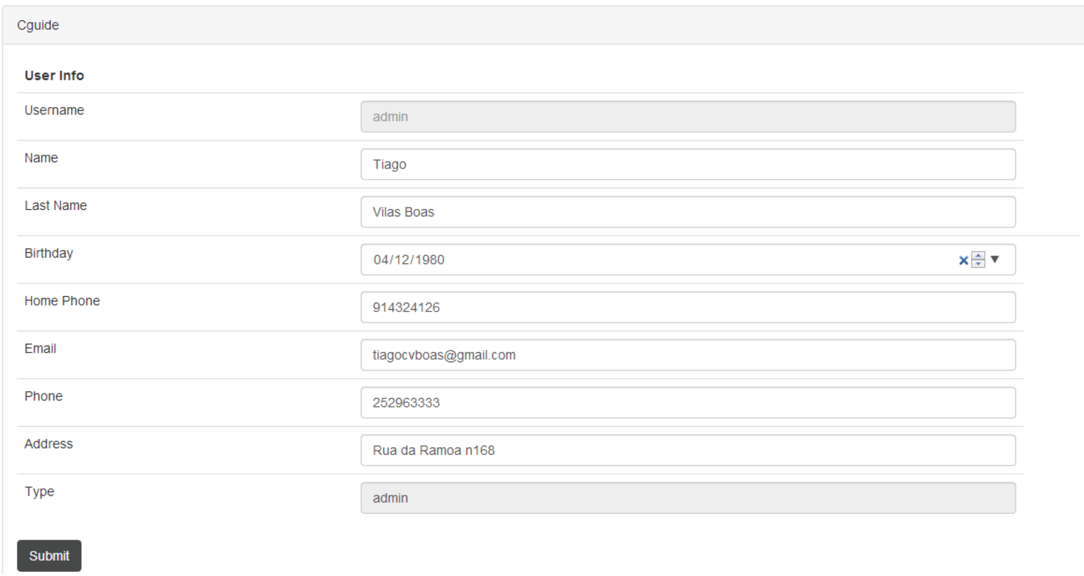

Figure 13 - User control panel.

3.6.2. Managing Guidelines

A user can list the available guidelines on the system, it also allows the user to check the information about the guideline such as description, date of creation, update and so on. The creation and edition is yet to be implemented, though the server side of creation and editing is already complete, but given the complexity of creating a guideline, the best way to create a web based interface to create and update the guideline is still under discussion. An administrator user will be able to view, execute, edit and add new guidelines, but a regular user, which will only be able to execute and view user information; because of the complexity of a guideline creation it should not be available to every user as seen on figure 14 and 15.

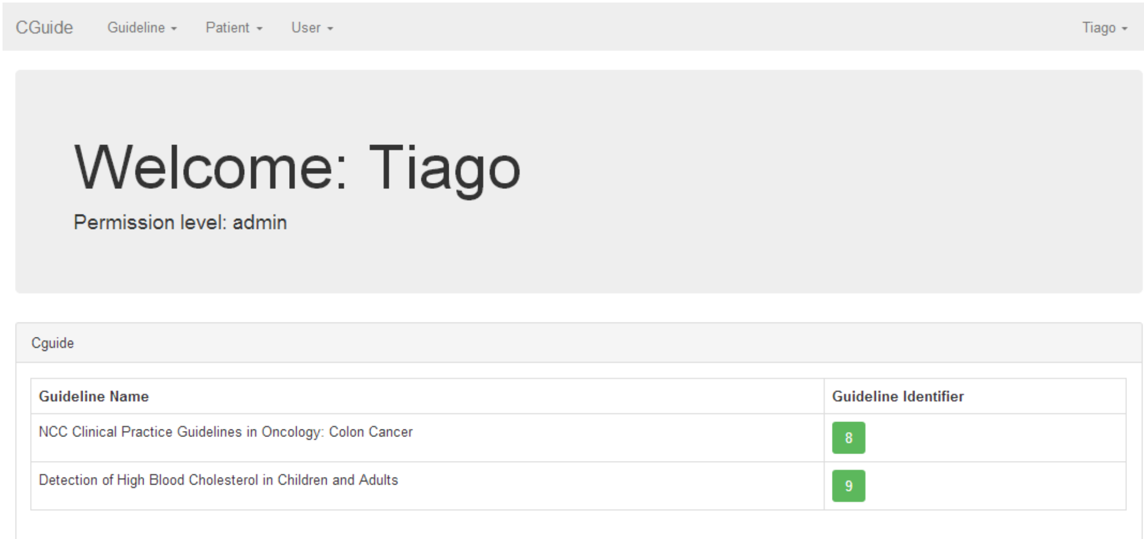

Figure 14- Guideline list using the web application.

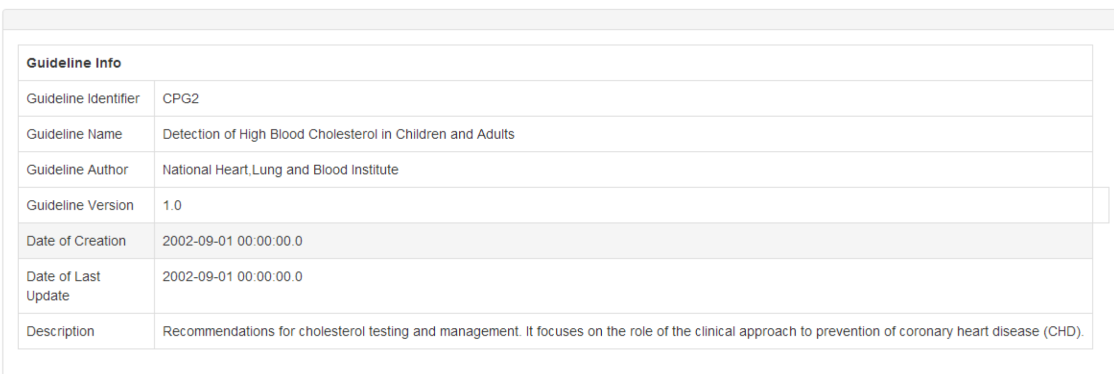

Figure 15 - Guideline details.

3.6.3. Managing Patients / Users

- Patients

The application offers the possibility of a user adding, removing, and editing patients to the patient database, every user is allowed to do it.

- Users

Users can be added, edited, listed and removed but only by an administration user, the only way to add a user is to manually add him to the system, due to the nature of the information dealt within the application, such as medical information, privacy and control of the users' information is essential.

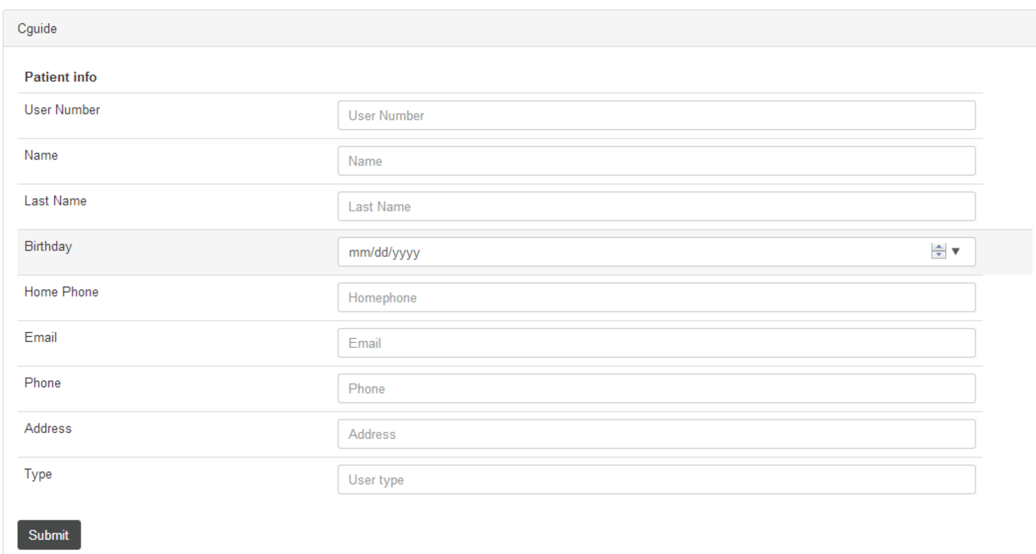

Figure 16 – Form to add a patient to the system to allow guideline execution.

3.6.4. Start/Resume Execution

There's the option for the user to start a new guideline execution as well as manage the old execution as seen of figure 18, given that some guidelines can have duration from hours to years, so a required feature was the ability to resume a specific execution. Whenever a task is executed the new state of the execution is store on the database, allowing the possibility to store at any point even if the session ends. This offers the ability to resume both on the web application or in the future the mobile application. Also allows that the user can resume a task even if using another computer.

When creating a new guideline execution, the user must introduce the patient number, and can set a description to it. It creates a new instance of the system with the current state that can be later resumed.

The user is allowed to remove and resume a specific guideline, though a guideline execution is never physically removed from the database, it just stays has invisible for the user, since data from that execution such as questions and decisions generated can be re-utilized on later executions.

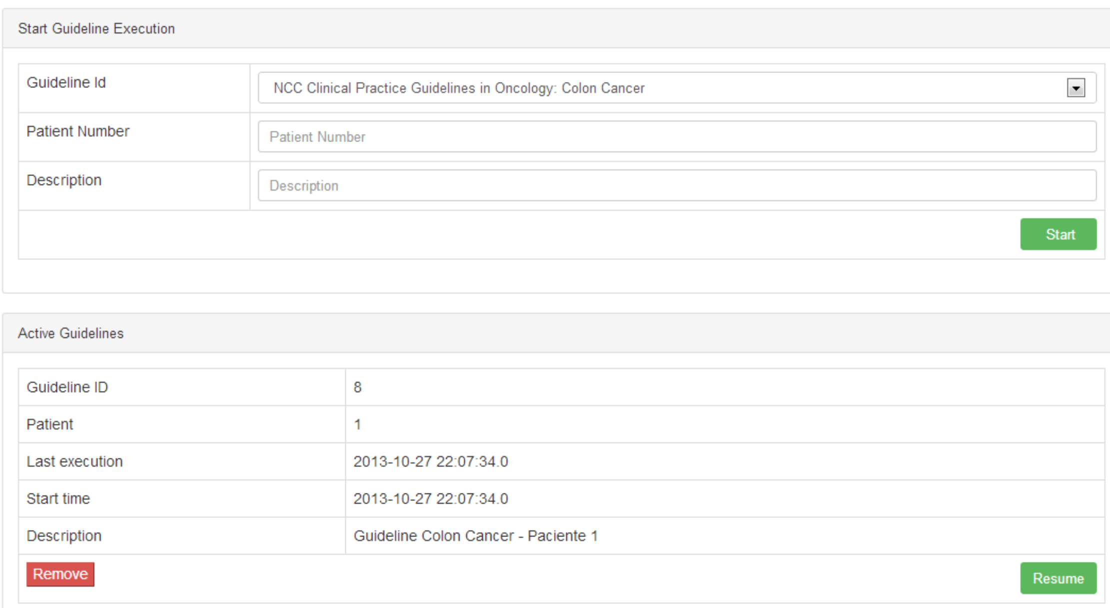

Figure 17 - Guideline Execution Start/Resume.

3.6.5. Execute Guideline

The interface for the guideline execution covers all the range of possible tasks, from decision, plans, questions etc., it offers the user the ability to choose either to execute or pass to the next task, being the difference that on the execution the information of that specific task is stored on the database, while when passing it is not, this was a recommendation of the physician who is offering guidance for the CompGuide project.

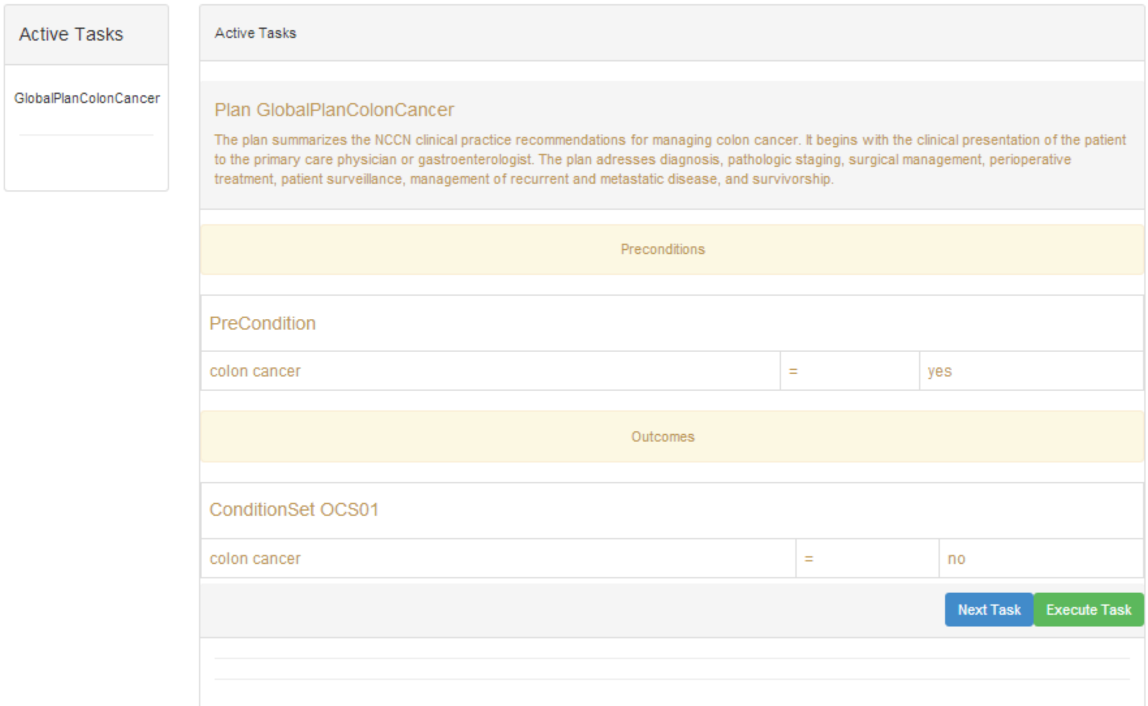

Figure 18 – Example of the beginning of execution of the plan Global Plan Colon Cancer.

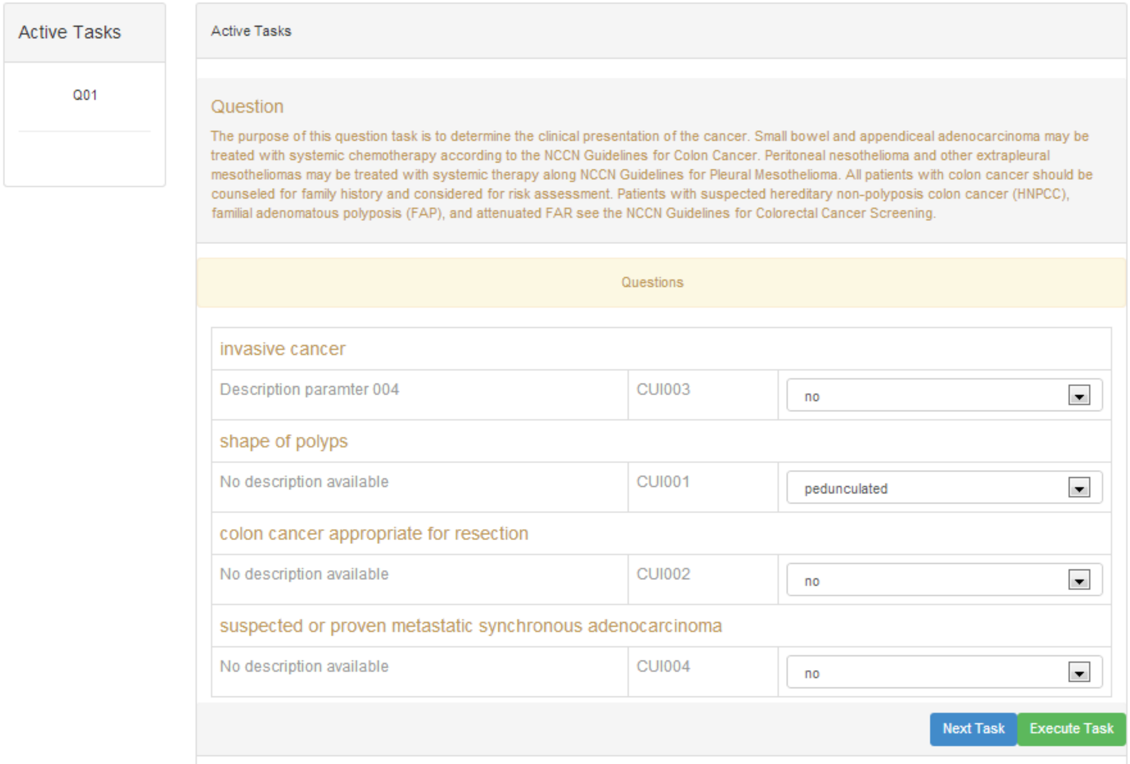

Figure 19 - Example of a Question pending to be executed.

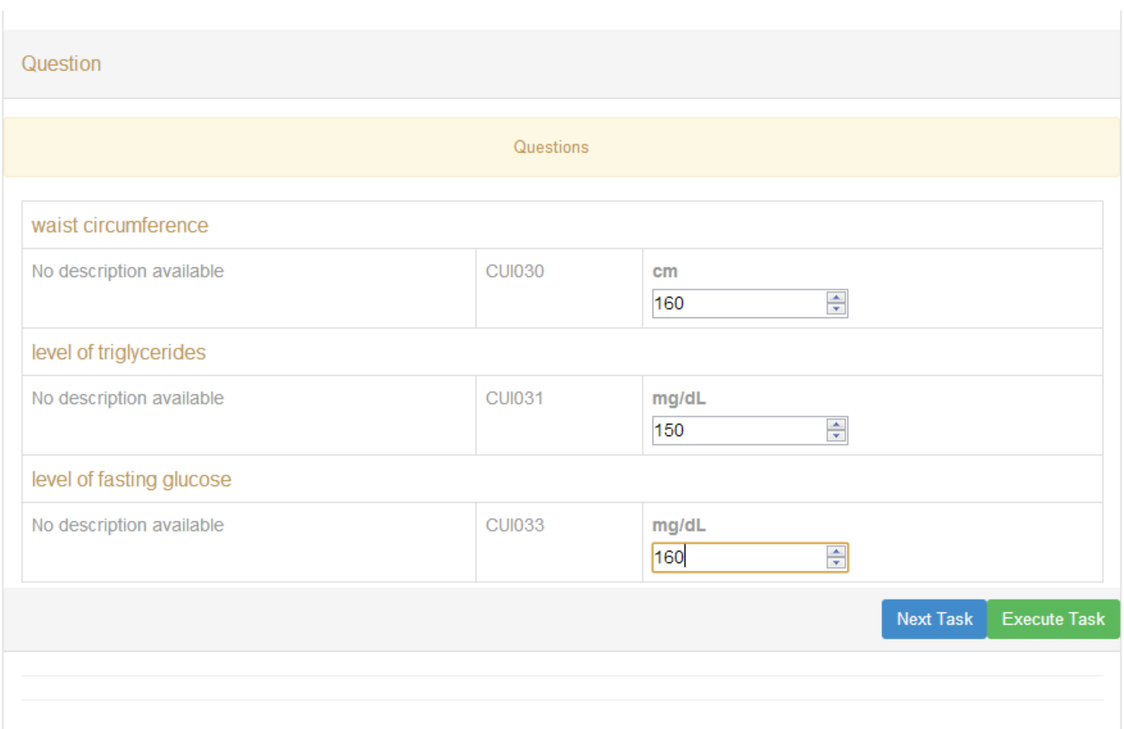

Figure 20- Execution of a Question with 3 different numeric parameters.

4. Creation and Execution of CIGs

Guideline execution is, eventually, the most complex component of this project, the execution posed a lot of problems while implementing the application, the details of which were not realized they until the development, issues such as temporal, clinical and task ordering constraints.

While the model of CIG representation is highly flexible to the creators, by enabling them to simulate any scenario, it created issues for the application development, the complexity of the application becomes bigger and bigger by every step of the execution. To handle this issue, all the computation required for the execution is done by the server.

4.1. Case Study

The set of case-studies used for testing the application's functionalities consisted of two CPGs, the National Comprehensive Cancer Network (NCCN) Guideline in Oncology for Colon Cancer[55] and the National Heart Lung and Blood Institute (NHLBI) Guideline for High Cholesterol[56].

The NCCN guideline was chosen for its complexity. It presents situations of concurrency and alternation of medical recommendations, with multiple temporal patterns and occurrences of different situations of conditions placed on tasks. This guideline was also selected due to its length, providing recommendations for diagnosis and treatment of colon cancer. Being a lengthy guideline also makes it more error prone when applied to real clinical setting, so this would be the ideal starting point for a real-life prospective evaluation of the application. Colorectal cancer is the third most common cancer and the fourth most frequent cause of cancer death worldwide[57]. North America, Australia and Western Europe are the regions with the highest incidence rates while developing countries have the lowest rates[57]. The disease develops in the cells lining the colon when they suffer mutations. In this situation, the mutations cause the uncontrollable growth of these cells and they begin to invade healthy tissues, yielding malignant tumours.

On the other hand, the NHLBI guideline was selected due to the different decision situations it contains. It is primarily used to safeguard situations of cardiovascular disease which is also one of the main causes of death in developed countries[58].

The execution of the different case studies offers the possibility to see some of the capabilities of the application to deal with unique situations, such as the ability to control temporal constraints, the ability to execute parallel tasks and stopping tasks. Parallel tasks required the development of a synchronization system. A small example of the execution of parallel tasks can be seen on Figure 21 and 22.

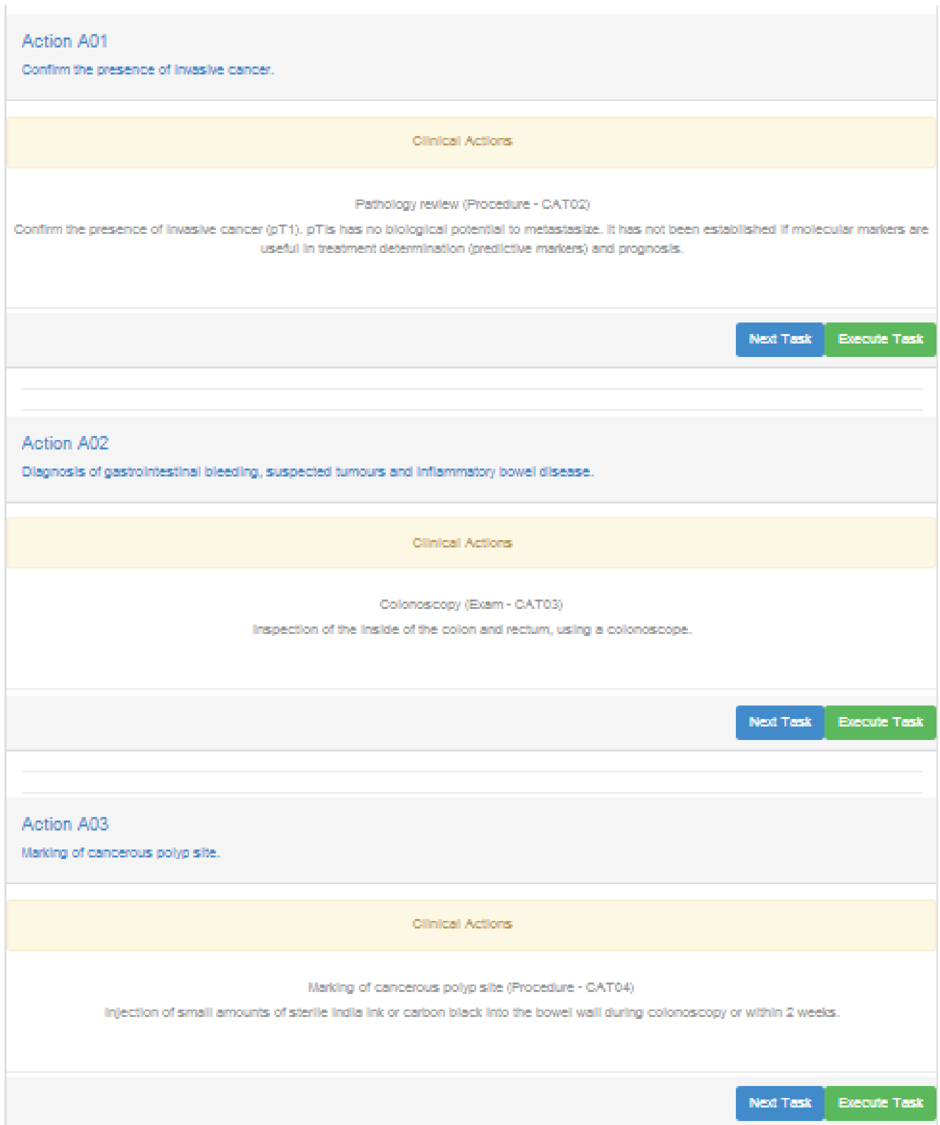

Figure 21- Executing parallel tasks

The application offers the user the possibility to execute the task it prefers. In this case there are three tasks to be executed, each leading to another task, and all of them eventually synchronize in the same task. All of the three tasks represented in Figure 21 proceed to the same synchronization task, though when executed like on Figure 22, the next task will stay hidden until all the other tasks reach it.

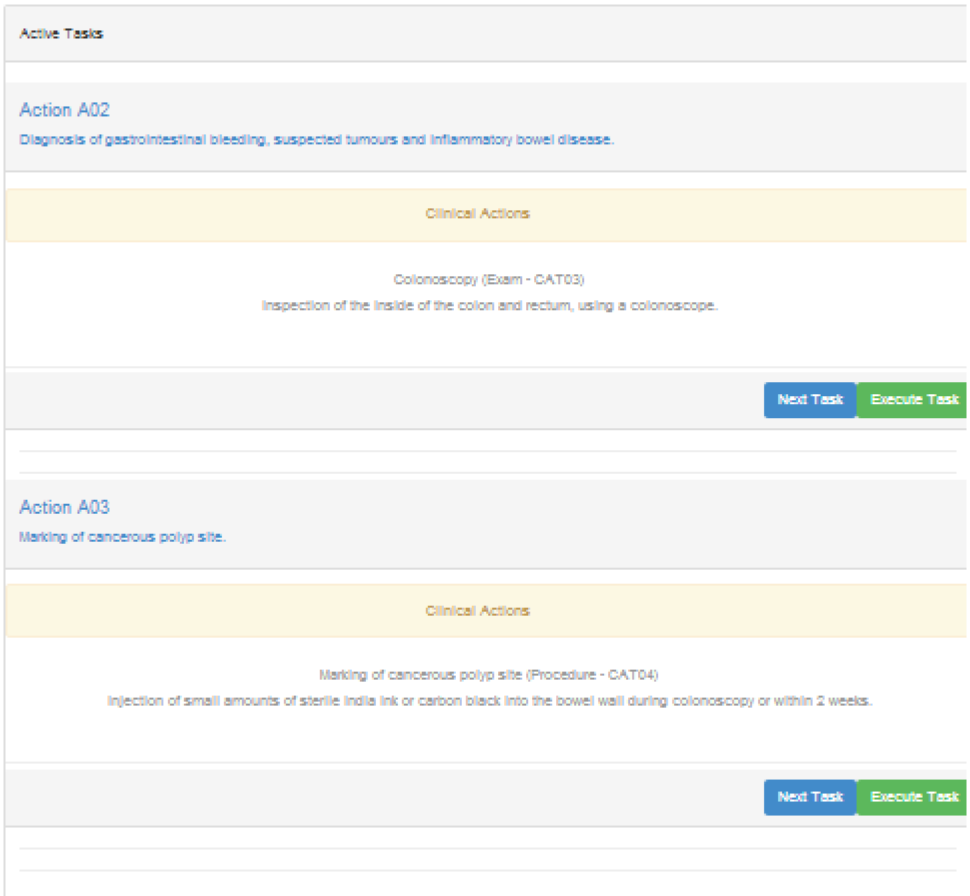

Figure 22 - After executing Action A01, that task reached the synchronization task and keeps hidden until all of them do so.

The application is also able to filter and decide on an outcome of a decision. It can process the data that came before and achieve a new patient state, which can be later used by other decisions or task selection process. The patient state is updated according to the outcome of the decision.

The next example shows some relevant aspects from the execution of the NHLBI guideline, hiding tasks like End and Plan which are irrelevant in this example. The application makes a series of questions as seen on Figure 23. When it receives the answers, the application provides the next task, which is a decision, as shown in Figure 24, that will be made based on the questions made before, causing a change in the state of a patient. Later, the decision is taken and the it will appear to the user as seen on Figure 25, alongside the next tasks, if there are any. The decision is then stored has a patient state usable for the next guideline executions with the same patient.

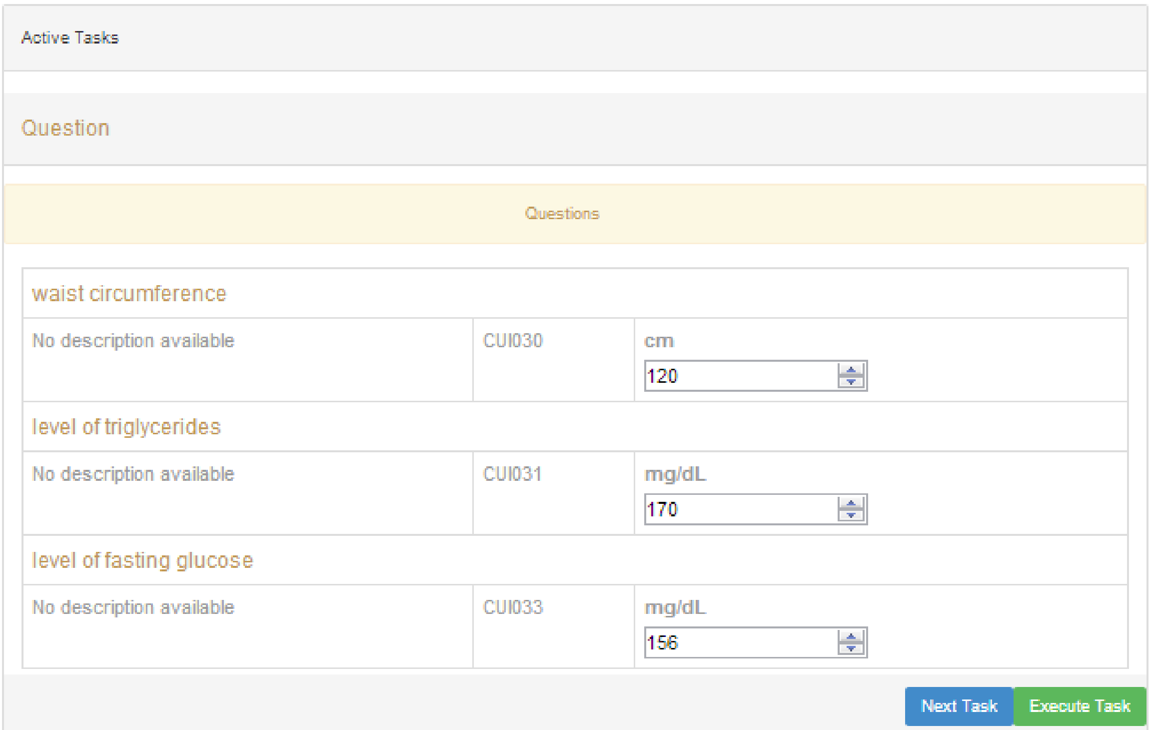

Figure 23 - Question to gather data for the decision to come

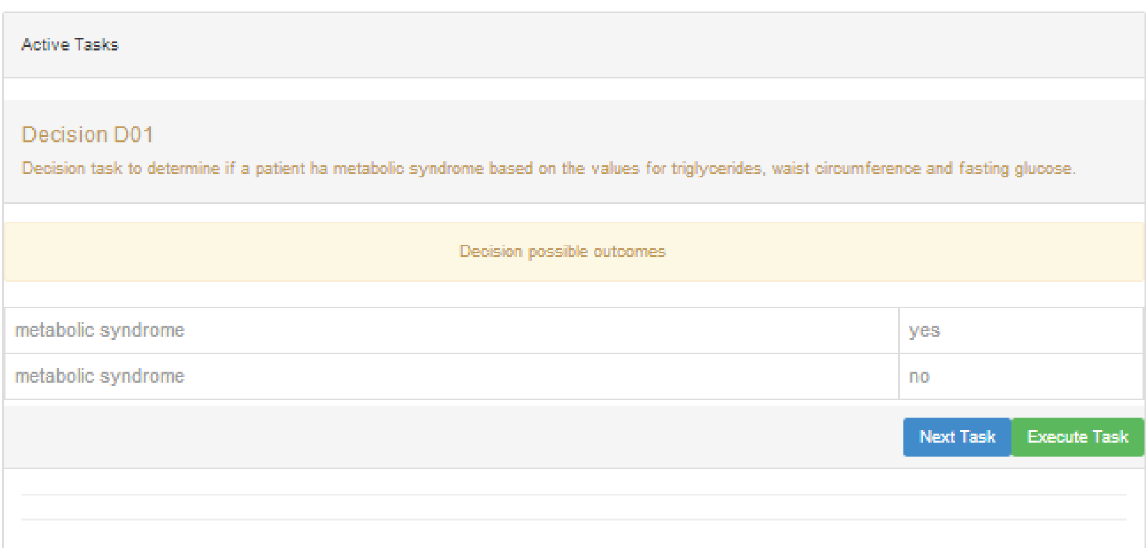

Figure 24 - Decision task with the possible outcomes

Figure 25 - The decision made by the application.

4.2. Creation

The creation can be achieved by a series of web services that were implemented to add, remove and modify every class that is defined in the guideline repository file in OWL. This functionality cannot yet be seen working due to the lack of a web interface to use it.

The creation of guidelines occurs by adding and removing specific individuals to the OWL file, the individual's identifiers are automatically generated by the application to avoid replication of identifiers, which would cause inconsistencies on the execution.

Individual reutilization is also achieved by simply using the created individual identifier when inserting a specific individual of any class.

4.3. Execution

Execution is a complex combination of user interface, task processing and computation, the idea of executing guidelines on the Web or any application is to reduce the amount of work for the clinical professional, by allowing him to focus on what is most important, the patient. It also works as a sort of task checker, to avoid forgetting a task.

It also allows the professional to deviate and even completely ignore it if he so desires because in some cases the guideline might not be exactly what the patient needs by choosing to cancel it, or simply skip to the next task as seen on figure 21.

The implementation of the guideline execution is achieved by tackling three things, selection and verification of task ordering and verification of clinical constraints.

The execution, if performed by the user on the server, consists in simply submitting the button and sending the identifier of the chosen *generatedTask* (a task that is possible to execute next), to the server by a specific web service. The server replies with the appropriate next task and tasks still pending, while controlling all the constraints required. The Response consists of the data of the task and, if a decision is made, a processed decision to be displayed.

4.3.1. Verification of Task Ordering

Task ordering is a challenge that appears by the fact that the guideline model allows any task to be of any kind of task, so as an example a parallel task, which means more than a task to be executed, can be followed by another parallel task, this causes issues with task controlling, because a parallel task requires a synchronization task to be executed eventually by all the spawned tasks.

To tackle all the issues found, a task controller was implemented that is constantly updated when a task is executed and stored to preserve the state of the guideline execution. The task controller consists of the following structures:

- A quadruple that stores the info of a task, it consists of the identifier of the created task (*Database Identifier*), the task identifier (*OWL Identifier* required to fetch info from the OWL file), the synchronization guideline identifier of the task(*OWL Identifier*), the identifier of the plan to which that task belongs(*Database Identifier*). This offers control over parallel tasks. By implementing a series of tests, these structures allow the control and storage of the of every specific detail of the state of the execution, allowing the control over the different plans execution and different parallel tasks, ensure that the correct synchronization is achieved.

- Parallel task queue, to store the pending parallel tasks, a queue keeps the order of the elements and always removes the last item of the list that was inserted, which will preserve the order of the tasks to synchronize first and control the task flow, by removing the tasks that are synchronized but still have sibling tasks pending, it stores the id of the task that came before it.[59]

- A list of identifiers of tasks that are alternative, to store when there is the option of executing one of a number of tasks, but only one can be executed. The list is then controlled by removing all of the sibling tasks whenever one of the alternative tasks is executed. A series of comparisons are made using the parent identifier and plan on the task quadruple to ensure that they belong to exactly the same parent and plan so that tasks are not wrongly removed.

- A queue to store the pending plans, because the CompGuide model allows a task to be a plan. Then for the same reason as the parallel task queue was implemented, there is the need to make sure the child plan that spawned later finishes first.

Some of the reasons for this kind of structure to be required are shown in Figure 26, as the figure shows, the next task N1 has three parallel tasks to execute which all synchronize on the same task, the N6 task, but P1.1, P1.2 and P1.3 all have different next tasks. Although N4 and N3 both have N6 as the nest task, raising no synchronization issues, N2 spawned new parallel tasks, P2.1 and P2.2. There are several issues that arise from a possibility like this, the first being that if a user executes P1.2, N3 will reach N6 before the other parallel tasks do, so N6 needs to wait for P1.2 and P1.3 to also reach N6 as the next task so that the user can execute N6. Another problem is the control over parallel tasks as a parallel task has to be synchronized before the task that was previously created for the simple fact that there is still a synchronization task pending from the older parallel task. This is the main reason for the quadruple, to be able to check if a task is already synchronized and if it is allowed to execute the next task.

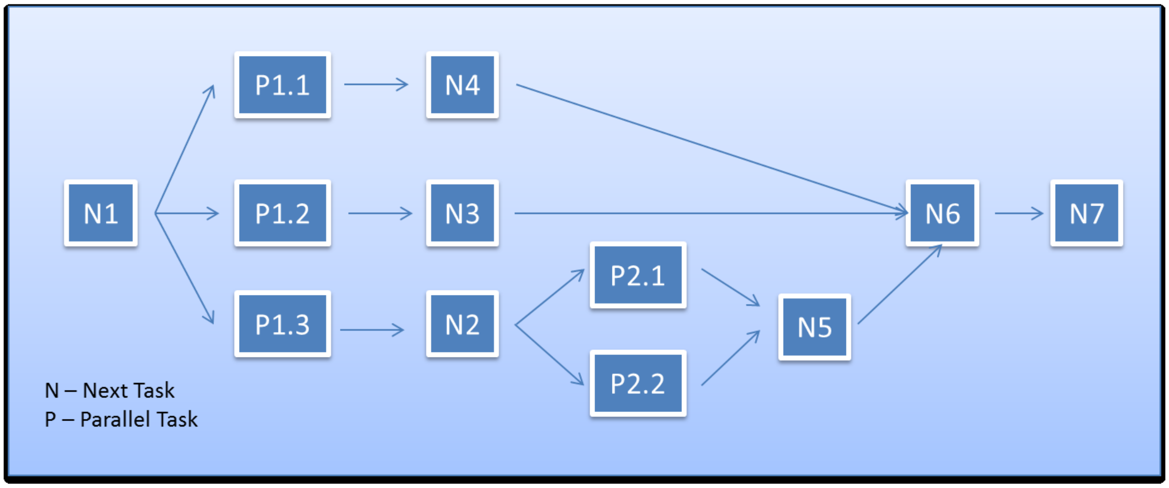

Figure 26 - Example of a parallel task synchronization problem.

Many controlling methods had to be implemented in order to control the execution; to add and remove elements, all had to be custom implemented to achieve the full control of the tasks. It is fully functional and it is working as expected.

4.3.2. Verification of Clinical Constraints

Some tasks require a set of conditions to be fulfilled before the task can be executed, it is the responsibility of the guideline creator to make sure the required information is retrieved by the use of the question tasks, which consist of getting information from the patient, whenever this information of a patient is collected, it is store on the database table Observation.

Whenever a clinical constraint is required, the system will query the database for the data of the patient stored on the observations, and use that information of the state of the patient to proceed with the computations to check if those conditions match, for every specific state of the patient. Whenever a new state of the patient is introduced that already exists, the database will update the field, instead of creating a new entry to avoid replication of data as much as possible.

4.3.3. Verification of Temporal Constraints

All the tables of the database contain a field named time, this is used to store the moment when the specific action, task, observation, etc., was collected, it allows us to use it to control temporal constraints. If a task cannot be performed until a specific time, tests will be made after converting the duration and interval of a specific task by matching that time with the time set on the database, to verify if a task is executable.

This offers the ability to control temporal constraints, tasks, or any other element of the guideline and it is essential to the scope of guideline execution.

4.4. Discussion

There were many challenges both technical and structural. Such challenges included how to handle authentication, what application server to use, mastering REST web services and its structures and how much to be processed on the server or the client. It became evident that the more information is processed on the server side, the easier and cleaner the development of the web application gets, also leaving the heavier processing tasks to the server which has more computing power then the regular client.

Controlling tasks became an issue in the first attempts to execute the colon cancer guideline, given its huge complexity, containing numerous tasks to be performed simultaneously. This eventually lead to a more complex control of the tasks, creating the possibility of incredible complex states of executing with different parallel tasks and plans even though these states will rarely happen, the execution engine is ready to handle them. Offering full compatibility with the CompGuide model.

When developing the core server it became clearer the amount of work required for an application of this scale, which forced the focus of development on the core-server to provide the required functionalities expected by a guideline execution application. Forcing the Web application to become a secondary priority as it acts as interface with the human side. The basic functions and full guideline execution interface were implemented, which were the core objectives of the application. This choice was made considering that those features can be added later and by just using the services provided already by the core-server, so the main focus was on providing those essential services such as proper task control, user authentication and authorization, question processing and storing, and a functional patient data model.

Although creation is one of the main features, the client side (Web application) was left for further development, but the server side is complete, allowing a future implementation of the Web application creation interface.

Comparing with the existing alternatives, this application offers a new approach over the existing applications, it uses a more expressive CIG model, so it offers features like parallel and alternative tasks which most of the other applications do not, the ability to reuse components of previous guidelines like plans or even specific tasks, it will allow faster development of guidelines by the guideline creators.

While the guideline model already offers more functionality and expressivity to guideline execution, the main difference and advantages of this application versus applications like GLEE, Arezzo, and others is the architecture of the application itself. Being a Service Oriented Application (SOA) it allows for easy development of new tools and custom client execution application by using the Web API created, it promotes scalability and simplicity of development while still offering a rich execution environment.

The chosen architecture and technologies offer a wide variety of advantages over the other applications, the core server can run both on Linux or Windows Servers, being coded in Java with only open Source technologies, the server can be installed in pretty much any computer available at no cost. The web application is also platform independent, it can run on every computer that has a Web browser installed and it is possible to publish. It allows remote guideline execution with data centralization, so it offers a new approach that is more suitable for a work environment.

The ability to easily expand and fix the application is also an advantage, adding features to the application can be done by simply adding new web services, adding and integrating outside resources is also easier since most of the resources nowadays are offered by using web services.

The main disadvantage is that the application is that it requires an internet connection to make full use of the capabilities. Being web based offers the ability to only requires the deployment on one computer and although it is possible to use it like a desktop application by running both the web application and the server on the same computer, that is not the purpose of such an application, but make it online and to be used from different places and devices and centralize all the date on the same database.

The possibility of developing a mobile application offers this project a wider range of interest, since mobile applications are easy to use, and can be used anywhere with a simple tablet or smart phone, it allows for practitioners to be mobile, while executing a guideline.

This application offers a modern architecture to a field which is growing, given the convenience of an assisted guideline execution process, while being as complete and sometimes more expressive than the current options available.

5. Conclusions and Future Work

CPGs present themselves as a challenging yet interesting field of study, the current solutions offer tools to create and execute CPGs. All of the solutions are focused still on the academic field by either lacking in functionalities such as time constraints, dependency of other services and high complexity for both users and CPG developers. Others, despite being filled with many functionalities and tools, are difficult to use, with complicated and non-intuitive interfaces making them less suitable for daily use in such a complicated environment such has health facilities. All the solution have a common fault, difficulty to deploy in a real environment by either being the result of using several different tools, or by simply not being designed to do so.

CompGuide is one of the first attempts to build such a platform, which can easily be deployed, easy to use and have all the required features for CPG execution. The architecture allows one framework to have all the functionalities thanks to the use of OWL ontology, JAVA and being Web-based. Being multi-platform and encoded in JAVA, makes it easy to expand to other ubiquitous platforms such as Android.

The objectives proposed were mostly achieved, the execution of guidelines using CompGuide model is now possible, not all the objectives were achieved though, the application still misses the integration with services like UMLS and creation of guidelines using the Web application. But the main objective of creating centralized easily expandable guidelines execution application was achieved is now possible to be able to execute multiple guidelines with multiple users from multiple places, and centralizing the information stored.

5.1. Synthesis of the Work Done

The implemented server features allow creation and execution of guidelines. The server is complete, offering all the required web services for all the features implemented on the web application. It also has all the web services to fetch data from every table on the database, given the permission of the user.

Authentication and authorization to control data access, so that sensible information cannot be easily retrieved by non-authorized entities. Control of the execution such as temporal, execution and clinical constraints are also fully implemented, and the tests of the different kinds of tasks performed using the two guidelines on the database, which contain most of the possible scenarios, show that the system is stable and almost fully functional.

5.2. Relevant Work

This application offers something the others do not, the ability to centralize the application, to provide a service instead of an application. Using it will not be limited to a local database of a specific terminal where the application is running, but a wide variety of users that will be able to contribute with creation, guidelines, and data of guideline execution that will be shared by all the users, this offers information sharing.

A global database, or EMR also provides the ability to constantly add new users and share the information wherever the location the execution is performed, offering a wider amount of data to later be used for data mining and statistical analysis.

The model itself, while highly expressive for guideline development, offered a wide variety of challenges to the development of the execution engine, given the architecture, it forced the development of a completely different method of storing the execution state, and being able, through a simple web service request, to restore, evaluate and execute a specific task, while finding the next task.

5.3. Future Work

The future work consists in extending the functionalities of the Web Application, expanding the range of services the core server can provide, since with the rich amount of data that is collecting during the execution of guidelines, many other features might show up if the system starts to get more testing and gets more guidelines on the guideline repository. But features are not a problem since they are easily implemented by extending the number and functionality of web services by the core server.

Integration with the UMLS, is also another of the objectives to be done in the future, to normalize the medical terms of the exams, medications and all the medical procedures, given the fact that UMLS offers web services to use their features it should be easy given this project's architecture to make use of the resources of this dictionary to normalize the terms used within the application.

The development of an android application is also one of the goals of this project, given the architecture of the whole system. The application should be easily implemented, since all the features are offered by the server, the android application would consist only on developing the user interface.

Proceed with more tests to assure the stability of the system and finding possible errors and bugs before deploying it for a wider use by clinical practitioners in order to test functionality.

6. Bibliography

[1] R. D. Zielstorff, "Online practice guidelines: issues, obstacles, and future prospects.," *Journal of the American Medical Informatics Association*, vol. 5, no. 3, pp. 227–236, 1998.

[2] W. R. Hersh, E. Care, and I. S. An, "Medical Informatics," vol. 288, no. 16, pp. 1955–1958, 2002.

[3] E. Coiera, "Guide to Health Informatics 2 nd Edition Chapter 25 - Clinical Decision Support Systems," pp. 1–12, 2005.

[4] "Relatório Mundial da Saúde FINANCIAMENTO DOS SISTEMAS DE SAÚDE."

[5] J. Zhang, V. Patel, and T. Johnson, "Medical error: Is the solution medical or cognitive?," *Journal of the American Medical …*, vol. 9, no. 6, pp. 75–77, 2002.

[6] C. De Melo, "Stress ocupacional em profissionais de saúde: um estudo com médicos e enfermeiros portugueses," vol. 14, no. 3, pp. 239–248, 2009.

[7] F. A. Sonnenberg and C. G. Hagerty, "Computer-Interpretable Clinical Practice Guidelines Where Are We and where Are We Going ?," pp. 145–158, 2006.

[8] C. M. Farquhar, E. W. Kofa, and J. R. Slutsky, "Clinicians�ttitudes to clinical practice guidelines: a systematic review," *methods*, vol. 3, p. 5, 1992.

[9] A. Toker, S. Shvarts, Z. H. Perry, Y. Doron, and H. Reuveni, "Clinical guidelines, defensive medicine, and the physician between the two," *American journal of otolaryngology*, vol. 25, no. 4, pp. 245–250, 2004.

[10] M. Crane, "Managed care 1999. Clinical guidelines: a malpractice safety net?," *Medical economics*, vol. 76, no. 7, pp. 236,239–243, Apr. 1999.

[11] B. C. James and M. E. Hammond, "The challenge of variation in medical practice.," *Archives of pathology laboratory medicine*, vol. 124, no. 7, pp. 1001–1003, 2000.

[12] H. Cho and J. Choi, "Ubiquitous Computing in Healthcare," vol. 45, no. December 2002, pp. 2002–2003, 2003.

[13] L. K. Field MJ, *Clinical Practice Guidelines: Directions for a New Program*. 1990.

[14] G. Eysenbach, "What is e-health?," *Journal of medical Internet research*, vol. 3, no. 2, p. E20.

[15] P. Drury, "The eHealth agenda for developing countries.," *World hospitals and health services the official journal of the International Hospital Federation*, vol. 41, no. 4, pp. 38–40, 2005.

[16] G. Eysenbach and A. R. Jadad, "Evidence-based Patient Choice and Consumer health informatics in the Internet age," *Journal of Medical Internet Research*, vol. 3, no. 2, p. E19, 2001.

[17] M. W. M. Jaspers, M. Smeulers, H. Vermeulen, and L. W. Peute, "Effects of clinical decision-support systems on practitioner performance and patient outcomes: a synthesis of high-quality systematic review findings.," *Journal of the American Medical Informatics Association*, vol. 18, no. 3, pp. 327–334, 2011.

[18] D. L. Hunt, R. B. Haynes, S. E. Hanna, and K. Smith, "Effects of computer-based clinical decision support systems on physician performance and patient outcomes: a systematic review.," *Jama The Journal Of The American Medical Association*, vol. 280, no. 15, pp. 1339–1346, 1998.

[19] C. S. Rand, N. R. Powe, A. W. Wu, and M. H. Wilson, "Why Don ' t Physicians Follow A Framework for Improvement," *October*, vol. 0718, no. 15, pp. 1458–1465, 1999.

[20] S. H. Woolf, R. Grol, a Hutchinson, M. Eccles, and J. Grimshaw, "Clinical guidelines: potential benefits, limitations, and harms of clinical guidelines.," *BMJ (Clinical research ed.)*, vol. 318, no. 7182, pp. 527–30, Feb. 1999.

[21] N. L. Rutten F, Brouwer W, *Practice guidelines based on clinical and economic evidence. Indispensable tools in future market oriented health care.* 2005.

[22] P. A. De Clercq, J. A. Blom, H. H. M. Korsten, and A. Hasman, "Approaches for creating computer-interpretable guidelines that facilitate decision support.," *Artificial Intelligence in Medicine*, vol. 31, no. 1, pp. 1–27, 2004.

[23] K. M. Hrabak, J. R. Campbell, S. W. Tu, R. McClure, and R. T. Weida, "Creating interoperable guidelines: requirements of vocabulary standards in immunization decision support.," *Studies In Health Technology And Informatics*, vol. 129, no. Pt 2, pp. 930–934, 2007.

[24] A. ten Teije, M. Marcos, M. Balser, J. van Croonenborg, C. Duelli, F. van Harmelen, P. Lucas, S. Miksch, W. Reif, K. Rosenbrand, and A. Seyfang, "Improving medical protocols by formal methods," *Artificial Intelligence in Medicine*, vol. 36, no. 3, pp. 193–209, Mar. 2006.

[25] P. A. de Clercq, J. A. Blom, H. H. M. Korsten, and A. Hasman, "Approaches for creating computer-interpretable guidelines that facilitate decision support," *Artificial Intelligence in Medicine*, vol. 31, no. 1, pp. 1–27, 2004.

[26] D. Isern and A. Moreno, "Computer-based execution of clinical guidelines: a review.," *International Journal of Medical Informatics*, vol. 77, no. 12, pp. 787–808, 2008.

[27] L. InferMed, "Arezzo Technical White Paper."

[28] C. Re- and O. Brookes, "The Syntax and Semantics of the PRO forma Guideline Modeling Language," vol. 10, no. 5, pp. 433–443, 2003.

[29] Y. Shahar, O. Young, E. Shalom, A. Mayaffit, R. Moskovitch, A. Hessing, and M. Galperin, "DEGEL: A hybrid, multiple-ontology framework for specification and retrieval of clinical guidelines," in *Artificial intelligence in medicine: 9th Conference on Artificial Intelligence in Medicine in Europe, AIME 2003, Protaras, Cyprus, October 18-22, 2003 proceedings*, 2003, vol. 2780, p. 122.

[30] Y. Shahar, O. Young, E. Shalom, M. Galperin, A. Mayaffit, R. Moskovitch, and A. Hessing, "A framework for a distributed, hybrid, multiple-ontology clinical-guideline library, and automated guideline-support tools.," *Journal of Biomedical Informatics*, vol. 37, no. 5, pp. 325–344, 2004.

[31] S. Miksch, Y. Shahar, and P. Johnson, "Asbru: A task-specific, intention-based, and time-oriented language for representing skeletal plans," in *Proceedings of the 7th Workshop on Knowledge Engineering: Methods & Languages (KEML-97)*, 1997, p. 9.

[32] P. Terenziani, S. Montani, A. Bottrighi, G. Molino, and M. Torchio, "Applying artificial intelligence to clinical guidelines: the GLARE approach.," *Studies in health technology and informatics*, vol. 139, pp. 273–82, Jan. 2008.

[33] P. Terenziani, S. Montani, A. Bottrighi, M. Torchio, G. Molino, and G. Correndo, *The GLARE Approach to Clinical Guideline: Main Features*, vol. 101, no. iii. IOS Press, 2004, pp. 162–166.

[34] M. Peleg, A. A. Boxwala, E. Bernstam, S. Tu, R. A. Greenes, and E. H. Shortliffe, "Sharable representation of clinical guidelines in GLIF: relationship to the Arden Syntax," *Journal of Biomedical Informatics*, vol. 34, no. 3, pp. 170–181, 2001.

[35] 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.

[36] J. Choi, L. M. Currie, D. Wang, and S. Bakken, "Encoding a clinical practice guideline using guideline interchange format: a case study of a depression screening and management guideline.," *International Journal of Medical Informatics*, vol. 76 Suppl 2, pp. S302–S307, 2007.

[37] D. Wang, M. Peleg, S. W. Tu, A. A. Boxwala, O. Ogunyemi, Q. Zeng, R. A. Greenes, V. L. Patel, and E. H. Shortliffe, "Design and implementation of the GLIF3 guideline execution engine," *Journal of biomedical informatics*, vol. 37, no. 5, pp. 305–318, 2004.

[38] D. Wang, M. Peleg, S. W. Tu, A. A. Boxwala, O. Ogunyemi, Q. Zeng, R. A. Greenes, V. L. Patel, and E. H. Shortliffe, "Design and implementation of the GLIF3 guideline execution engine.," *Journal of Biomedical Informatics*, vol. 37, no. 5, pp. 305–318, 2004.

[39] M. Wooldridge, *An Introduction to MultiAgent Systems*, vol. 86, no. 3. John Wiley & Sons, 2002, p. 348.

[40] K. E. Campbell, D. E. Oliver, and E. H. Shortliffe, "The Unified Medical Language System: toward a collaborative approach for solving terminologic problems.," *Journal of the American Medical Informatics Association : JAMIA*, vol. 5, no. 1, pp. 12–6.

[41] D. Isern, A. Valls, and A. Moreno, "Using aggregation operators to personalize agent-based medical services," in *10th International Conference on KnowledgeBased Intelligent Information and Engineering Systems KES 2006*, 2006, vol. 4252, pp. 1256–1263.

[42] D. Berg, P. Ram, J. Glasgow, and J. Castro, "SAGEDesktop: an environment for testing clinical practice guidelines.," *The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 2, pp. 3217–3220, 2004.

[43] T. Oliveira, "Active guidelines in a clinical setting," 2012.

[44] T. Oliveira, P. Novais, and J. Neves, "Representation of Clinical Practice Guideline Components in OWL Advantages of Ontology Web Language."

[45] "Web Services." [Online]. Available: http://en.wikipedia.org/wiki/Web_service. [Accessed: 15-Apr-2013].

[46] "REST." [Online]. Available: http://en.wikipedia.org/wiki/Representational_state_transfer. [Accessed: 14-Aug-2013].

[47] "Web API." [Online]. Available: http://en.wikipedia.org/wiki/Web_API. [Accessed: 24-Aug-2013].

[48] M. A. Casteleiro and J. J. Des Diz, "Clinical practice guidelines: A case study of combining OWL-S, OWL, and SWRL," *Knowledge-Based Systems*, vol. 21, no. 3, pp. 247–255, Apr. 2008.

[49] "OWL - API." [Online]. Available: http://owlapi.sourceforge.net/. [Accessed: 15-Apr-2013].

[50] "JBoss Application Server." [Online]. Available: http://www.jboss.org/jbossas. [Accessed: 25-Mar-2013].

[51] "JSON." [Online]. Available: http://en.wikipedia.org/wiki/JSON. [Accessed: 16-Apr-2013].

[52] "Gson." [Online]. Available: https://code.google.com/p/google-gson/. [Accessed: 17-Apr-2013].

[53] "Twitter Bootstrap." [Online]. Available: http://en.wikipedia.org/wiki/Bootstrap_(front-end_framework). [Accessed: 12-Aug-2013].

[54] N. Versi, R. Pablo, and P. Guti, "Introducción," 2010.

[55] NCCN, "NCCN Clinical Practice Guideline in Oncology Colon Cancer," 2013.

[56] NHBLI, "Detection, Evaluation, and Treatment of High Blood Cholesterol in Adults," 2004.

[57] Globocan, "Colorectal Cancer Incidence and Mortality Worldwide in 2008," 2008. [Online]. Available: http://globocan.iarc.fr/factsheets/cancers/colorectal.asp. [Accessed: 16-Oct-2013].

[58] WHO, "Cardiovascular diseases (CVDs)," 2013. [Online]. Available: http://www.who.int/mediacentre/factsheets/fs317/en/index.html. [Accessed: 15-Oct-2013].

[59] "Queue." [Online]. Available: http://en.wikipedia.org/wiki/Queue_(abstract_data_type). [Accessed: 15-Aug-2013].