

Computer-Based Management of Clinical Guidelines: A Survey

David Isern¹ and Antonio Moreno

Abstract. *Clinical guidelines* are useful tools to standardize and improve health care. The automation of the steps involved in a guideline is a basic step towards their widespread use in medical centres. This paper presents a survey and a comparison of the main current projects and approaches that are trying to define and implement systems that allow the automation of clinical guidelines. One of the main results of the comparison of these methods is that agent technology is arguably one of the best options to implement this kind of systems.

1 Introduction

Clinical guidelines (CGs) are a very active area of research at the moment. CGs contain a set of directions or principles to assist the health care practitioner with patient care decisions about appropriate diagnostic, therapeutic, or other clinical procedures for specific clinical circumstances. Medical guidelines are intended to ensure consistent high quality clinical practice and provide some benefits to both patients and health managers ([8, 12]), such as:

- To facilitate reuse, because a guideline can be adapted, tailored and applied to different clinical situations.
- To support rapid dissemination of updates and changes.
- To use the clinical knowledge about the patient at the appropriate point of his/her care.
- To improve clinician performance and patient outcomes (e.g. a clinician will not forget an important aspect to be checked before following a certain treatment).
- To encourage guideline authors to employ rigorous formal techniques, that will help to ensure syntactic, logical and medical validity ([29]).

Although the use of guidelines gives some benefits to both patients and practitioners, and also several international organisations create and maintain some repositories with guidelines in different domains such as cancer, general practitioner, or pediatrics, they are not being widely used. In [2] some factors that limit or restrict a complete physician adherence to a clinical guideline were identified. Such factors, called *barriers*, were organised into groups based on whether they affected physician knowledge, attitude or behavior. The authors of that survey identified different barriers associated with each group, such as lack of awareness, lack of familiarity, lack of agreement, lack of self-efficacy, lack of outcome expectancy, inertia of previous practice, external factors (e.g. inability to reconcile patient preferences with guideline recommendations) and environmental factors

(e.g. lack of time or lack of resources). These factors could be tackled (in most cases) with an automation and computerisation of the daily management of both clinical guidelines and patient data.

The main goal to be accomplished in any guideline-based health-care system is to improve patient care. Several steps must be considered in the use of medical guidelines: representation, acquisition, verification and execution aspects. The first three tasks concern the authors of the guideline, whereas the later is related to practitioners. Let us briefly consider these steps:

- Representation.* A CG contains several elements to be modelled, such as actions, required patient data, decisions to be taken, constraints between tasks, temporal constraints in a global planning, etc. Nowadays, different researchers have defined formal languages to model computer-interpretable clinical guidelines, such as PROforma, EON, GLIF or Asbru [5].
- Acquisition.* Medical guidelines are based on the evidence collected from clinical trials and existing literature [23]. Some authors are also currently working in the semi-automatic construction of guidelines, by applying Machine Learning techniques to the clinical data collected in a medical centre about the treatment given to a particular set of patients [27].
- Verification.* Verification includes two aspects: is a medical guideline well formed?, and, which of these two available medical guidelines is the best? The first question is addressed to verify the formal correctness of the guideline [29]. The second question is more difficult because it is necessary to quantify how good is a medical guideline. To tackle this problem, some authors proposed a methodology called AGREE, that calculates a set of parameters for a given medical guideline in order to evaluate its *quality* [24].
- Execution aspects.* As commented above, a medical guideline contains a lot of information to be considered (decisions to be taken, constraints between tasks, temporal restrictions). All these data have to be collected and monitored when enacting the guideline.

This paper is focused on the analysis of systems that allow the automatic (or semi-automatic) execution of guidelines. There are several systems that are currently being developed for this purpose. The more relevant are the following: *SAGE*, *GLARE*, *GLEE*, *NewGuide*, *Arezzo*TM, *DeGeL*, *SpEM*, and *HeCaSe2*. In the next section we define the basic characteristics to be analysed in a guideline execution engine (e.g. coordination issues, security techniques, use of standard medical vocabularies) and then each of the eight systems named above is summarily described. Sect. 3 provides a table that summarizes the main information about each system and comments the results obtained for each of the analyzed attributes. Finally, the last section states some conclusions extracted from the comparison of the systems.

¹ Universitat Rovira i Virgili (URV), Department of Computer Science and Mathematics, Artificial Intelligence Research Group (BANZAI). Tarragona, Catalonia (Spain). Email: {david.isern,antonio.moreno}@urv.cat

2 Guideline execution engines

A guideline-based execution engine should ideally fulfil the following requirements:

- keep a repository of guidelines,
- facilitate the creation of guidelines through a graphical editor, or even define a methodology to create or reuse guidelines,
- provide a formal language for encoding medical guidelines,
- provide mechanisms to coordinate the services required in the use/management of guidelines,
- allow the user to analyse the behaviour of the guideline (e.g. by providing a run time engine or a simulator),
- provide a connection with an Electronic Health Record (EHR),
- allow the use of standard vocabularies inside guidelines, and,
- provide security to both transmissions and storage of sensitive data related to patients.

In the next subsections we analyse the eight guideline-based tools named in the introduction, and in the next section the issues listed above are summarised and compared.

2.1 GuideLine Acquisition and Execution (GLARE)

GLARE is a domain-independent system that can deal with clinical guidelines [30, 31]. *GLARE* distinguishes between the *acquisition phase*, when a guideline is introduced in the system (e.g. by a committee of experts), and the *execution phase*, when a guideline is applied by physicians to a specific situation (i.e. it is instantiated on a given patient).

Internally, CGs in *GLARE* do not use any standard representation. Their authors have defined a proprietary graph-based representation, where each action is represented by a node (different types of actions are available), while control relations are represented by arcs. They define three layers called *System*, *XML* and *DBMS* (see Fig. 1). The *System Layer* contains the two modules mentioned above, called *acquisition* and *execution*. The lower level, called *DBMS Layer*, connects physically the higher levels with databases where all required data for both creating and executing guidelines is stored. There is data about available resources, terminology used in guidelines, information about drugs, information about all open instances of guidelines, a repository of guidelines, and a patient's medical record. Moreover, *GLARE* defines an intermediate layer called *XML Layer* that allows to represent/manage/exchange data between the *DBMS Layer* and the *System Layer* in a structured way [30].

The authors distinguish between *atomic* and *composite* actions. *Atomic* actions are simple actions to be performed in a particular point of the guidelines. Three possible atomic actions were defined: *i*) *queries*, that allow to request any external information, *ii*) *work actions*, that represent actions to be performed, and *iii*) *decision actions*, that embed a criteria to select an alternative among a set of actions that could be performed at a certain point. On the other hand, *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. The execution engine maintains the current state of all actions and monitors all their preconditions before starting them.

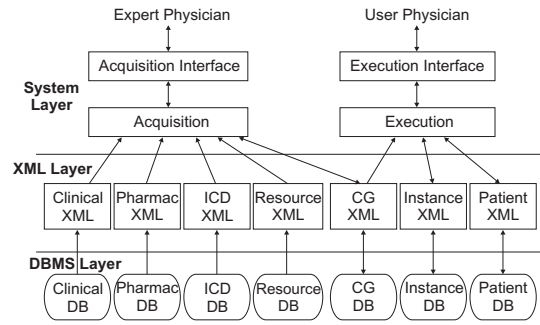


Figure 1. *GLARE* general architecture

This system is focused in the management of *temporal constraints* between different tasks in a CG, and allows to execute/simulate a CG using the appropriate retrieved data (*execution module*). Each patient has its own medical record (contained in the *Patient DB*), which is updated continuously with the actions executed through a CG. The architecture is complemented with a database of available resources in a given hospital (*Resource DB*), that allows to make domain-dependent execution of guidelines. Moreover, a contextualisation module has been added to *GLARE*. It allows the local adoption and update of guidelines to cope with both the need to apply them to new situations (countries, hospitals and/or departments), and with the need to manage updates (e.g. authoring, recording the history of a guideline and learning from experience).

2.2 Standards-Based Sharable Active Guideline Environment (SAGE)

The *SAGE* project is a collaboration among research groups at six institutions in the US [33, 32]. The ultimate goal of the project is to create an infrastructure that will allow execution of standards-based clinical practice guidelines across heterogeneous clinical information systems. The global architecture of the system, that includes a guideline execution engine as its central component, is shown in Fig. 2. It is important to note the integration of this engine with current clinical applications, and also the definition of a central core of terminologies (models that detail not only medical data but also patient data, care workflow processes and the structure of health care organisations).

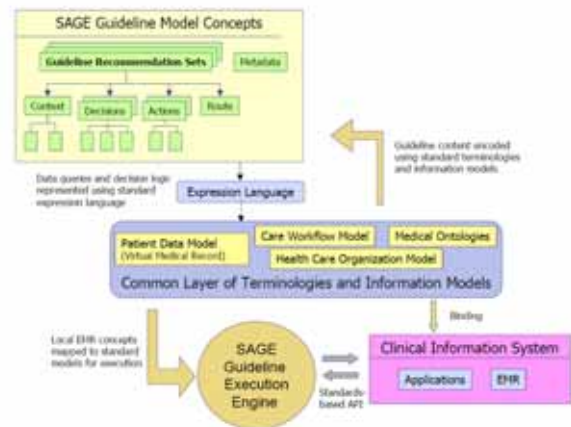


Figure 2. *SAGE* global architecture

The *SAGE* project has created a guideline model with the following features:

- 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 CGs in order to manage all available guidelines.
- It uses standards to represent the data (electronic medical record and processes) such as SNOMED-CT and Health Level 7 (in particular, HL7v3) [18]. The internal representation of guidelines is made using the EON formalism [34].
- It includes organizational knowledge to capture workflow information and resources needed to provide decision support in enterprise settings. It proposes a methodology to develop/create medical guidelines.

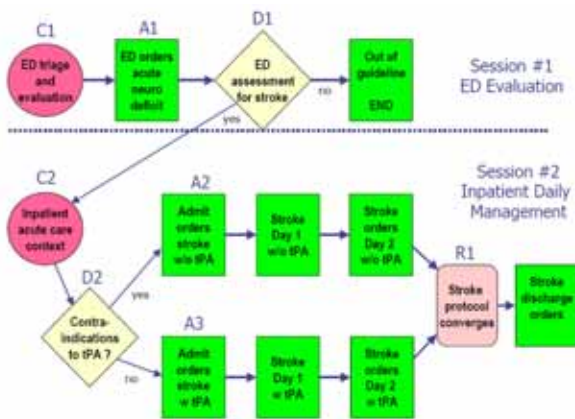


Figure 3. The top-level process specification in a *SAGE* guideline.

Fig. 3 shows a portion of a *SAGE*-defined guideline and how the elements should react to the events in the care process. *SAGE* defines two different formalisms: *recommendation-set* and *decision-map* [33]. The *recommendation-set* is an activity graph composed by processes and interactions between them. Activity graphs allow the specification of computational algorithms or medical care plans as processes consisting of *i*) contexts, that are combinations of a clinical setting (e.g. outpatient visit in a general internal medicine clinic), care providers to whom the recommendation is directed, relevant patient attributes (e.g. patient age), and possibly a triggering event (e.g. a patient checking into the clinic), *ii*) decision nodes, that evaluate conditions on variables (e.g. a Boolean precondition for an action), *iii*) action nodes, that encapsulate a set of work items that should be performed either by a computer system or by a healthcare provider, and *iv*) routing nodes, that are used purely for branching and synchronization of multiple concurrent processes. In the figure shown above, *C1* and *C2* represent context nodes, *A1*, *A2* and *A3* are action nodes, *D1* and *D2* are decision nodes, and *R1* is an example of a routing node.

A *decision map* consists of a collection of decisions, each of which contains a context (similar to those in an activity graph), a collection of action choices, and a decision model to determine the appropriate choice in each possible circumstance.

2.3 GLIF3 Guideline Execution Engine (GLEE)

GLEE is a tool for executing guidelines encoded in the 3rd version of GLIF (called GLIF3), which has been developed across different institutions as the Department of Biomedical Informatics (Columbia University, US), the Stanford Medical Informatics Lab. (Stanford University, US), the Decision Systems Group (Brigham and Women's Hospital, Harvard Medical School, US), the Department of Management Information Systems (University of Haifa, Israel), and Eclipsys Corporation (Boston, US) [35].

In addition to serving as an interface to the GLIF3 ([1]) guideline representation model, *GLEE* defines interfaces to connect it with electronic medical records (EMRs) and other clinical applications to facilitate its integration with the clinical information system at a local institution.

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. GLIF3 represents guidelines as flowcharts of temporally ordered nodes called guideline steps, and store actions (called *Action_Step*), decisions (called *Decision_Step*), and clinical states (called *Patient_Clinical_State*). It defines two more nodes, called *Branch_Step* and *Synchronization_Step*, which are used for modelling multiple concurrent paths through the guideline. Decision criteria are modelled using an OCL-based language (Object Constraint Language) called GELLO [1].

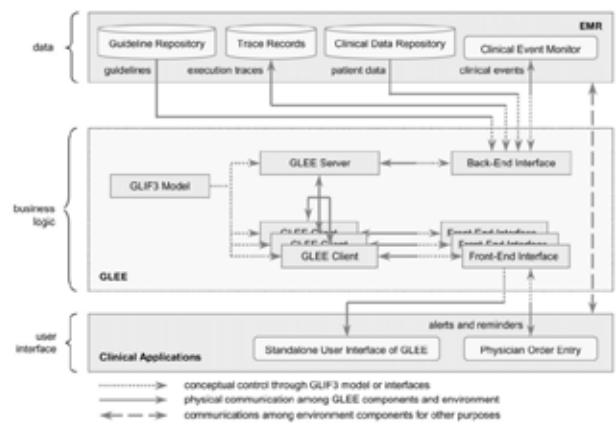


Figure 4. *GLEE* general architecture

As shown in Fig. 4, three levels of abstraction are defined: *data*, *business logic* and *user interface*. The *data* level contains the EMR with a guideline repository and the *clinical event monitor*, that allows the execution (or simulation) of clinical guidelines through an event-driven model. The *business logic* level contains the *GLEE* execution engine formed by a server and many clients. The server interacts with the data level, and clients interact with users (both through defined interfaces). At the lowest level, we find the *user interface* level where the clinical applications that exchange data with the upper levels are located.

GLEE has adopted two open standards: it uses Resource Description Framework (RDF) ([17]) as the exchange syntax for guidelines, and Health Level 7 (HL7) ([18]) to incorporate standard medical terminologies.

2.4 NewGuide

NewGuide is a framework for modelling and executing clinical practice guidelines (CGs) developed at the Laboratorio di Informatica Medica, Università di Pavia, Italy [3, 4].

NewGuide proposes the use of the Unified Medical Language System (UMLS, [18]) as a standard for terminology and a Medical Text Mark-up (MTM) language called *Guideline text mark-up* for describing tasks within a guideline [16]. Guidelines are represented using a representation language called GUIDE, which is based on Petri Nets. It allows to model complex concurrent processes as well as temporal, data and hierarchical issues [26].

GUIDE is integrated into a workflow management system which proposes an infrastructure that enables inter- and intra-organisational communication through a *Careflow Management System* (CfMS) that, on the basis of the available best practice medical knowledge, is able to coordinate the care providers activities. The final goal of this architecture is to provide Health Care Organizations (HCO) with technical solutions which should enable them to improve process efficiency, outcomes and quality of care.

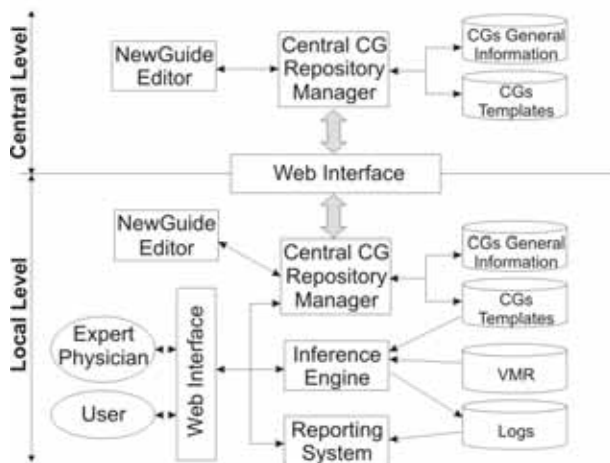


Figure 5. *NewGuide* general architecture

The CfMS is a system that defines, creates, and manages the execution of careflows (Cfs) through the use of software, running on one or more Cfs engines, which is able to interpret the care process definitions, interact with Cfs participants and, where required, invoke the use of ICT tools and applications. *Careflow* indicates the automation of a care process, in whole or in part, during which information, documents or tasks are passed from one participant to another for action, according to a process definition. Thus, GUIDE is an intermediate step, oriented towards medical experts, by means of which clinical guidelines may be formalised. A translation from GUIDE's objects to Workflow Process Definition Language (WPDL) is performed automatically. The system uses Oracle Workflow as a middleware layer to represent the electronic patient record, the consequent possibility of gathering information from different legacy systems, and the extension of the *virtual medical record* (VMR) to the storage of process data.

The inference engine is composed of a general manager, a message manager, and an instance manager (see Fig 6). As soon as a user asks for a CG from the *NewGuide* repository, the general manager creates an instance manager, which will enact an instance of that CG (*i.e.* a

CG referring to a particular patient). The instance manager interprets the GL flow and generates suggestions on the basis of the information stored in the VMR. The communication between *NewGuide* and the external world is governed by the message manager, which delegates requests and responses to the web user interface or to an external entity (through a SOAP interface) on the basis of the system configuration. The responsibility for maintaining the correct GL flow and timing (*i.e.* the subsequent activation) is left to the external CfMS.

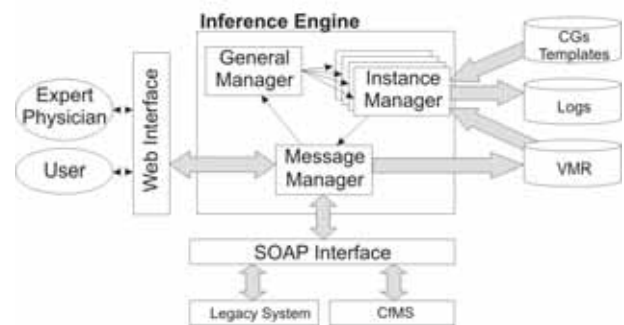


Figure 6. *NewGuide* inference engine

NewGuide authors have studied in detail the concept of *non-compliance with guidelines*. In [25] they analysed different factors that can cause a doctor not to follow a certain procedure; for example, a guideline will not provide the best recommendations for all patients under all possible circumstances, or a guideline can be applied in different ways depending on the clinical setting. For those reasons, guidelines should be evaluated *on the field* in order to assess both their applicability and the effectiveness of their implementation. This analysis can be a useful exercise because, according to the type of the detected non-compliance, improvements may be achieved by different interventions, such as site-specification of the guideline, users education, healthcare administrators involvement, and organisation re-engineering.

2.5 Arezzo™

Arezzo™ is a commercial decision support and guideline technology tool that uses *PROforma* as the basis [9, 10]².

The tool is composed by several engines: a *composer*, a *tester* and a *performer* (see Fig. 7). The *composer* is used to create guidelines using *PROforma* as representation language. The *tester* is used to test the guideline logic before deployment. The *performer* inference engine can then run the guideline, taking into account data related to patients stored in an electronic medical record (EMR).

PROforma is an executable process modelling language that has been successfully used to build and deploy a range of decision support systems, guidelines and other clinical applications. It is one of a number of the available proposals for representing clinical protocols and guidelines in a computer-interpretable format [22]. That language offers a declarative interchange 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 a third party, as a doctor or

² Two main implementations of a *PROforma* engine are currently available: the *Arezzo™* implementation, which is commercially available from InferMed Ltd, and the *Tallis* implementation developed at Cancer Research UK.

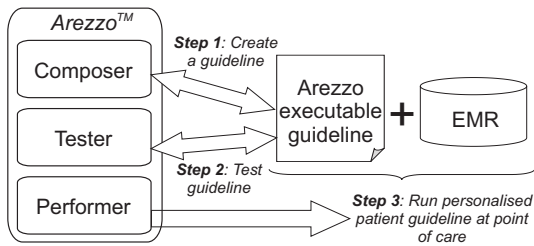


Figure 7. Arezzo™ architecture

a 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. During the enactment of a guideline in the performer engine, a task changes its value attributes and its internal state depending on whether the task is awaiting for any data, suspended, finished, or it cannot be accomplished in a specified point.

Arezzo™ uses the Domino autonomous agent model ([10]). The model deals with a large class of medical problems and establishes a relationship between decision making and plan enactment procedures. The main goal of this model is to identify the main resources required in any language to represent clinical guidelines that can be used for both decision making and plan management. Fig. 8 shows the whole model, which is divided in two parts: the left side concerns the decision-making processes, and the right hand side is related to planning and scheduling of tasks. The process begins by taking into account a set of patient data. According to the model (step 1) we then propose a set of possible causes of the health problem (step 2) and identify a possible set of solutions (Step 3) with its associated arguments pro and con. At this point, the doctor can identify a disease that has to be handled (step 4), and the cycle is started again to treat this specific disease. If, at this point, we know the appropriate option to follow, we must continue by managing the selected therapy plan (step 5). The component steps of the plan will be scheduled (step 6), resulting in the execution of actions. An action will often produce postconditions that change the patient’s state (step 7). That new information can produce new goals to be managed with other therapies. This is a cyclic model that produces a sequence of decision-making and scheduling steps [11].

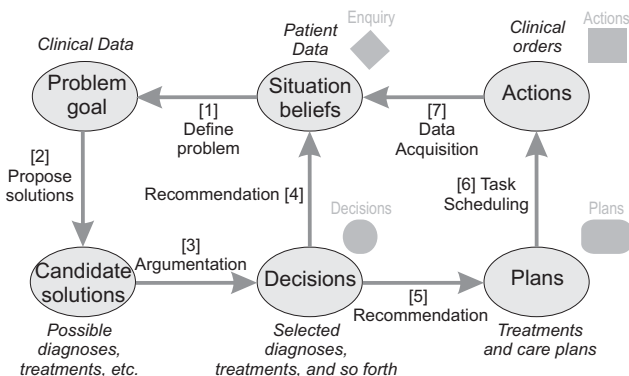


Figure 8. Generalised Domino model

2.6 Digital Electronic Guideline Library (DeGeL)

DeGeL is a Web-based, modular and distributed architecture, which facilitates the gradual conversion of clinical guidelines from text to a formal representation in a chosen language (such as Asbru) [28]. DeGeL contains a set of tools that support guideline classification, semantic mark-up, content-sensitive search, browsing, run-time application, and retrospective quality assessment (see the architecture of the system in Fig. 9).

The system maintains a repository of guidelines, and it allows the user to search, browse, retrieve and visualise all available guidelines. At the moment, the system creates guidelines using Asbru [5], 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.

The system uses different standards to represent the clinical information: LOINC-3 for observations and laboratory tests, ICD-9-CM for diagnosis codes, and CPT-4 for procedure codes.

The authors have designed some tools to solve different issues. A tool called *Uruz* allows practitioners or medical experts to create new medical guidelines. Another tool called *IndexiGuide* facilitates guideline retrieval. *VisiGuide* allows browsing and visualising guidelines. The run-time application, called *Spock*, is under development [36]; it incorporates an inference engine that can retrieve data stored in an EHR. The *Spock* guideline application engine consists of: *i*) a set of classes, that allow to store any guideline, *ii*) a parser, for interpreting the content of a guideline, *iii*) and a specialized module, the Controller, which synchronizes the communication between the system layers and external services.

Spock proposes an asynchronous method to monitor all actions made in a guideline. The method, called *application log*, stores different data structures, like the state transitions of a plan instance, a queue of scheduled awaiting tasks, and the list of recommended steps issued during application.

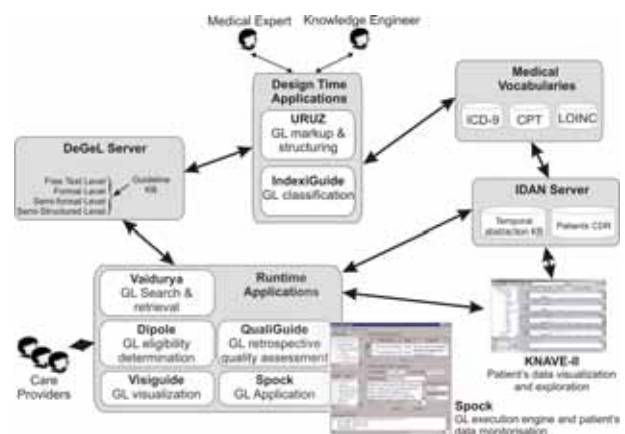


Figure 9. DeGeL general architecture

2.7 Specification Execution and Management Plan (SpEM)

Specification Execution and Management Plan (SpEM) is a framework for supporting the management of clinical guidelines [6, 7]. The authors defined a model that allows to both create and execute clinical guidelines. First of all, a general purpose language called PLAN was adapted to represent clinical guidelines. That language adopts an ECA (event-condition-action) approach, based on rules. This ECA rule mechanism is mapped into an existing DBMS, that is at the end who performs the enactment of a specified guideline through rising and managing different triggers. The execution module embedded in a DBMS uses these rules to start a guideline and, according to the raised events, activate a specified task at each point of time. An ECA rule is composed by three elements: *a)* an *event* part, containing a so-called transition predicate that lists all possible events which are of concern to the rule (it constitutes the situation that the rule has to monitor), *b)* a *condition* part, which can be an arbitrary predicate, and *c)* an *action* part, which is an arbitrary list of executable functions. That ECA rule paradigm contains the compositional primitives for any clinical guideline.

The system is divided in three main components: *specification plane*, *enforcement (execution) plane* and *manipulation plane* (see Fig. 10). The first module is able to capture clinical guidelines in a formal way. It provides methods to access, store and manipulate guidelines represented using PLAN [7]. The *enforcement plane* provides methods and tools for easing the creation and execution of patient-centred guidelines adapted from guidelines stored in the repository. The last module, the *manipulation plane*, provides facilities for querying and operating on guideline information through a high level language defined by authors called TOPSQL.

SpEM defines the following primitives that are required in a guideline or protocol representation model: *a)* an *action*, which represents any clinical or administrative task that is recommended to be performed, maintained, or avoided during the process of guideline application, *b)* a *decision*, that is a selection from a set of alternatives based on predefined criteria in a guideline, *c)* a *patient state*, which is a materialisation of a treated individual's clinical status based upon the actions that have been performed and the decisions that have been made, *d)* and an *execution state*, which is a description of a guideline current state [6].

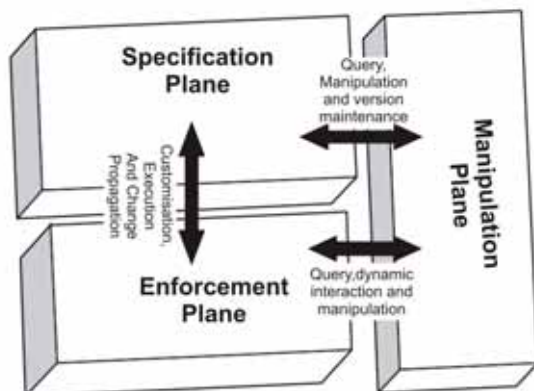


Figure 10. *SpEM* framework

2.8 Health Care Services (HeCaSe2)

HeCaSe2 is an agent-based platform that offers health care services to users (patients and doctors). The platform defines an architecture of agents with different roles, and with multiple interactions between them and humans or medical devices [13, 14].

The proposed architecture can be deployed around a network, in which each agent can act autonomously with its own knowledge and data. There isn't any central control and the number of agents depends on the specific configuration (*e.g.* doctors, departments, and devices in a medical center) [19].

The basic architecture of the MAS which is being developed in this work is shown in Fig. 11. At the top of the architecture is placed the user, who interacts with the system through his *User Agent (UA)*. This agent stores static data related to the user (*e.g.* national health care number, name, address, access information -login, password, and keys-) and dynamic data (the timetable and the preferences of the user). The *Broker Agent (BA)* is an agent that knows about all the medical centres located in a certain area. A *Medical Centre Agent (MCA)* centralises and monitors the outsiders accesses to the agents that manage the information of a medical centre. A MCA monitors all of its departments, represented by *Department Agents (DAs)*, and a set of general services linked to human or physical resources, represented by *Service Agents (SAs)* (*e.g.* a blood test service). Each department has a staff of several doctors, modelled through *Doctor Agents (DRAs)*, and offers more specific services, also modelled as SAs (*e.g.* a nurse that can take different observations *in situ*). Both MCAs and DAs are aware of the services they can provide (when a SA enters the system, it sends a message detailing its services to the associated MCA or DA). In addition, each department contains a *Guideline Agent (GA)* that performs all actions related to guidelines (*e.g.* it can retrieve the CG associated to a specific illness). This GA contains only CGs associated to the department where it is located. At the bottom of the architecture, a *Medical Record Agent (MRA)* controls the access to a database that stores all medical records of the patients of the medical centre. Appropriate security measures have been taken to ensure that only properly authenticated and authorised agents may access and update the medical records (see [20] for more details). Medical guidelines are enriched with a widely used terminology, the Unified Medical Language System (UMLS), in order to unify the vocabulary and promote the share of guidelines.

Clinical guidelines (CGs) are represented using *PROforma* ([11]). The coordination of tasks to be performed in a specific time is made in the DRA by retrieving all information concerned to the patient and following the execution of a guideline applied to a specific patient. Using the CG, a doctor can consider all the available data and take 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 ([15]). The medical services needed in the execution of a guideline can be located in the same medical centre or in another one (found with a previous discovery process). When the service is completed, the results are sent automatically from the service agent that has made the task to the MRA, that stores all patient's medical records. The interesting point is that all the scheduling processes and the follow-up of a guideline can be made automatically (or semi-automatically, with the doctor checking and confirming all relevant details) by agents in a autonomous way, without the patient having to waste time and effort to make a particular booking for each needed test or examination, or the doctor having to worry about asking to a service when the results of a certain patient will be available.

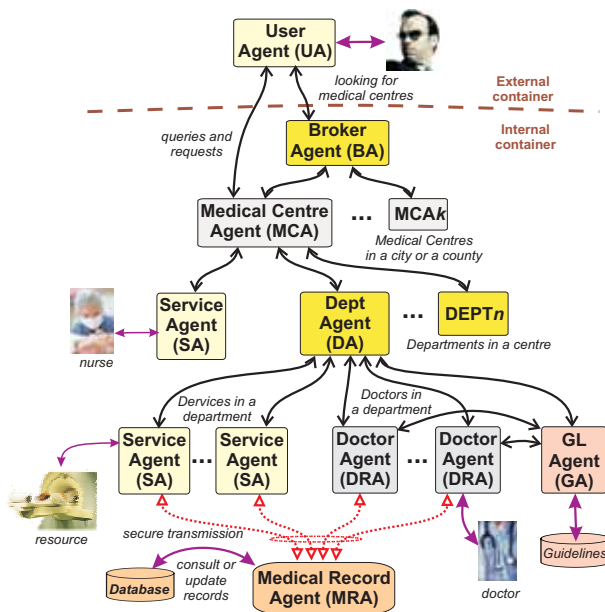


Figure 11. *HeCaSe2* agent-based architecture

3 Comparison

In previous sections, several guideline-based applications have been summarily described. All of them have different scopes, representations and architectures, according to the research interests of each developing group. This section provides a high level comparison of those tools, focused on the items that appear in Table 1 on the next page. Those items are: *a*) the existence of a repository of guidelines, *b*) if the tool offers a (graphical) editor to create and visualise their own guidelines, *c*) the language used to represent the clinical guidelines, *d*) if the tool is designed to be deployed as a multi-agent system, *e*) if the execution engine allows complex coordination elements such as parallelism, negotiation or scheduling, *f*) the existence of an execution engine, *g*) the connection of the system with an EHR, *h*) the use of any standard terminology defined by third parties, and finally *i*) the inclusion of security tools to preserve data integrity and authenticate the accesses to the (very sensitive) medical data exchanged in those systems.

Repository of guidelines

That element is offered by all tools. It allows to use the best available guideline at each moment and to update them when it is necessary. In some cases (*GLARE*, *HeCaSe2*, *NewGuide*) the repository stores several versions of a GL, allowing versioning. This feature is quite useful, as it can be used to update the GLs or to tailor a general GL to different centres according to the available resources in each location.

Guideline editor

The use of an editor to create guidelines is a recommended tool to ease the visualisation and updating of guidelines. Most of the described tools offer an editor, that can translate the clinical guidelines into the chosen representation language. Most of them identify several basic components (tasks, decisions, queries) which are linked together in a flowchart-based approach.

Only *SpEM* and *HeCaSe2* lack this module. The first one translates GLs directly to database commands, and the second uses a third party tool (Tallis) as editor.

Language used to represent the computer-interpretable clinical guidelines

There are several available languages to represent guidelines [22]. This is an important drawback because it prevents researchers from implementing tools in the same way, and there isn't any *de facto* standard language. As summarised in Table 1, all platforms define its own language or representation structures, according to their specific goals.

Agents

Although most of the platforms do not talk explicitly about agent technology or multi-agent systems, most of the architectures show how different autonomous components exchange information between them in order to retrieve and compose all required data in an enactment engine. In particular, *GLARE*, *GLEE* and *NewGuide* define different types of decentralised systems. *HeCaSe2* and *Arezzo*TM define an explicit agent-based architecture with different belief, desires and intentions for each entity.

Coordination

Clinical guidelines define different tasks to be accomplished. In any runtime engine it is very important to coordinate these tasks efficiently in order to improve the general performance. For instance, some tasks to be performed could have a time constraint (deadline) that has to be considered before tackling other tasks with more priority; at this point, the system must perform a booking between the patient and the resource that manages the required task. These complex tasks require coordination, negotiation and scheduling between all entities. Only two systems, *SpEM* and *HeCaSe2*, provide this type of facility.

Runtime engine

A runtime engine is required to simulate the behaviour of a clinical guideline with the patient data values. There are two approaches to perform the simulation: an event-based approach, such as *SpEM* and *GLEE*, and a rule-based approach, such as *NewGuide*, *GLARE* or *Arezzo*TM.

Access to EMR

There is a large number of different representations of Electronic Medical Records, but in all cases a gateway between any guideline-based system and an EMR is required to retrieve the required data in each moment. A knowledge base is usually needed to know exactly which attribute is required and to allow the system to find it within the EMR. Most of the tools implement an interface that allows the communication with a proprietary EMR representation.

Standards used

Medical terms can change significantly their meaning with little syntactic changes. A solution for that problem is the use of a standard terminology. In [18] there is an analysis of a set of the current available terminologies, including the ones used in the tools analysed in this paper. One of the most widely used terminologies is UMLS, that is a compendium of existing terminologies such as SNOMED, MESH, LOINC or ICD. The use of a terminology is an advantage in order to improve sharing and automation of the execution of clinical guidelines under any representation.

Table 1. Summary of guideline-based execution engines

<i>Tool</i>	<i>GL Repository</i>	<i>GL Editor</i>	<i>GL Repr. Lang.</i>	<i>Agents</i>	<i>Coordination</i>	<i>Run Time Engine</i>	<i>Access to EHR</i>	<i>Standards used</i>	<i>Security</i>
<i>GLARE</i>	Yes	Yes	Graph-Based	No	No	Yes	Yes	XML	No
<i>SAGE</i>	Yes	Yes	EON	No	No	Unclear	Yes	HL7, SNOMED	No
<i>GLEE</i>	Yes	Yes	GLIF3	No	No	Yes	Yes	HL7, RDF	No
<i>NewGuide</i>	Yes	Yes	GUIDE	No	No	Yes	Yes	UMLS	No
<i>Arezzo</i> TM	Yes	Yes	PROforma	Yes	No	Yes	No	No	No
<i>DeGeL</i>	Yes	Yes	Asbru	No	No	Yes	Yes	ICD9, CPT, LOINC	No
<i>SpEM</i>	Yes	No	PLAN	No	Yes	Yes	No	No	No
<i>HeCaSe2</i>	Yes	No	PROforma	Yes	Yes	Yes	Yes	UMLS	Yes

Security issues

Medical data is sensitive and has to be managed accurately. Transmissions, accesses and storage need a secure handling. To ensure secure transmissions, a ciphering of contents has to be made. [20] described an agent-based secure method to transmit contents based in a public key infrastructure. To ensure secure accesses, only registered (and authenticated) agents would be permitted to exchange information with any other agent of the system. Finally, the data should be stored in a secure database with authentication controls of the agents and users that want to access it. Of the analysed systems, only *HeCaSe2* implements explicit security features.

4 Conclusions

In this survey we have described the basic aspects of several applications oriented towards the automation of clinical guidelines. The result of this study is that, in this research area, there are several limitations that must be tackled in the future. One of them is the representation of computer-interpretable guidelines. The language of representation is the basis of these tools, and it could be desirable to adopt one formalism as standard and promote the interoperability between different tools and systems. This standardisation seems quite feasible, as most of the representation languages commented in this survey share the same basic components: some kind of action/decision/enquiry nodes, some mechanisms for coordination or synchronicity of actions, the ability to create sub-plans or sub-guidelines (so that different levels of abstraction can be considered when working with guidelines), or the possibility of storing the state of a guideline which is being executed.

Another important element of that automation is the existence of an Electronic Medical Record. The problem of finding the best possible representation of an EMR has not been solved yet, and applications are made *ad hoc* to fit a certain representation. That also limits heavily the interoperability of different tools.

It is also fair to say that in most countries (even the ones considered to be more technologically advanced) health care is not yet fully computerised, and nowadays it is not feasible (or it is hard and expensive) to include automated guideline enactment systems in real clinical settings. Most of the described systems consider as a big con-

cern the seamless integration of the execution of guidelines with the usual workflow of activities within a medical centre, in order to make it feasible to introduce this kind of systems in daily clinical practice. We considered the possibility of including another column in table 1, reflecting the actual use of each tool in clinical practice, but it seems that none of them is actually in daily use in any medical centre (as far as we have been able to find out, only *Arezzo*TM has been used for certain limited tasks).

As said in the comparison section, although most of the papers do not use the terms *agent* or *multi-agent system*, the authors of the analysed systems propose distributed architectures (usually as a client-server approach) with both data and tasks deployed around a computer network. Some authors propose an event-driven approach, that is similar to the communication-based approach that is the basis of multi-agent systems. Moreover, some of the proposed modules act autonomously, for instance runtime engines, and they could be easily mapped into agents. The *HeCaSe2* and *Arezzo*TM experiences, currently being developed, show how a guideline enactment system can be developed using an agent-based perspective with a formal language such as *PROforma*, that has been designed to interact with external elements linked to agents. This perspective allows to design more flexible and interoperable platforms that could be extended, in a feasible way, with more services and resources as required in a specific clinical setting. This extensibility and flexibility is especially clear in the *HeCaSe2* system.

Clinical guidelines include sets of rules that a doctor can follow in a specific situation (diagnosis, treatment, or prognosis). Coordination between humans and resources according to these rules is required to follow a guideline in a coherent way (ensuring the satisfaction of all relevant constraints). In an agent-based approach all participants in this global coordination are modelled as agents, and a set of communication-based algorithms can be applied, including distributed planning, coordination protocols or negotiation if a cost in the allocation of resources is considered. In a centralised model, this kind of coordination protocols are difficult to implement or the amount of data to be exchanged could suppose a bottleneck that could hinder system performance. We think that this is an important issue that shows the appropriateness of agent technology for the development of guideline enactment systems.

Some authors have argued in the last years that agent technology is especially suitable to design and implement systems in the health care domain [21]. As the final result of this comparative survey, we would like to argue that the basic well known characteristics of autonomous agents and multi-agent systems (spatial distribution, modelling of autonomous entities, use of dynamic coordination and planning procedures, etc.) make them a very appropriate technology to be used in guideline enactment systems. To emphasize this argument, the following table provides a tight mapping between the properties of intelligent agents and the requirements of that kind of systems.

Table 2. Mapping from guideline enactment systems to agents

<i>CG enactment</i>		<i>Agents</i>
Data distributed physically	⇔	Management of distributed knowledge, distributed execution of agents
Coordination of medical tasks	⇔	Communication-based protocols, dynamic coordination procedures
Independent entities	⇔	Autonomous agents
Scheduling medical visits/tests	⇔	Negotiation-based protocols, contract net mechanisms
Human-centered services	⇔	Interface agents, use of recommendation/preferences
Re-use of existing systems (e.g. CG repository)	⇔	Wrappers
Privacy of medical data	⇔	Agent authentication, message codification

ACKNOWLEDGEMENTS

The authors want to acknowledge the valuable comments made by Dr. Richard Thomson (editor of openclinical.org). This work has been partially supported by the IST project K4CARE (IST-2004-026968).

REFERENCES

- [1] A. A. Boxwala, M. Peleg, S. W. Tu, O. Ogunyemi, Q. Zeng, D. Wang, V. L. Patel, R. A. Greenes, and E. H. Shortliffe, 'GLIF3: A Representation Format for Sharable Computer-Interpretable Clinical Practice Guidelines', *Journal of Biomedical Informatics*, **37**, 147–161, (2004).
- [2] M. D. Cabana, C. S. Rand, N. R. Powe, A. W. Wu, M. H. Wilson, P. Abbound, and H. Rubin, 'Why don't physicians follow clinical practice guidelines? A framework for improvement', *Journal of the American Medical Informatics Association*, **282**, 1458–1466, (1999).
- [3] P. Ciccarese, E. Caffi, L. Boiocchi, S. Quaglini, and M. Stefanelli, 'A guideline management system', in *11th World Congress on Medical Informatics, MedInfo 2004*, eds., M. Fieschi, E. Coiera, and Y.-C.J. Li, volume 107 of *Studies in Health Technology and Informatics*, pp. 28–32, San Francisco, USA, (2004). IOS Press.
- [4] P. Ciccarese, E. Caffi, S. Quaglini, and M. Stefanelli, 'Architectures and Tools for innovative Health Information Systems: the Guide Project', *International Journal of Medical Informatics*, **74**, 553 – 562, (2005).
- [5] P. A. de Clercq, J. A. Blom, H. Korsten, and A. Hasman, 'Approaches for creating computer-interpretable guidelines that facilitate decision support', *Artificial Intelligence in Medicine*, **31**, 1–27, (2004).
- [6] K. Dube, *A Generic Approach to Supporting the Management of Computerised Clinical Guidelines and Protocols*, Ph.D. dissertation, Institute of Technology, Dublin, Ireland, 2004.
- [7] K. Dube, E. Mansour, and B. Wu, 'Supporting Collaboration and Information Sharing in Computer-Based Clinical Guideline Management', in *18th IEEE Symposium on Computer-Based Medical Systems (CBMS05)*, Dublin, Ireland, (2005). IEEE Press.
- [8] P. L. Elkin, M. Peleg, R. Lacson, E. Bernstam, S. W. Tu, A. Boxwala, R. A. Greenes, and E. H. Shortliffe, 'Toward Standardization of Electronic Guideline Representation', *MD Computing*, **17**, 39–44, (2001).
- [9] J. Fox, A. Alabassi, V. Patkar, T. Rose, and E. Black, 'An ontological approach to modelling tasks and goals', *Computers in Biology and Medicine*, *to appear*, (2006).
- [10] J. Fox, M. Beveridge, and D. Glasspool, 'Understanding intelligent agents: analysis and synthesis', *AI Communications*, **16**, 139–152, (2003).
- [11] J. Fox and S. Das, *Safe and Sound*, AAAI and MIT Press, 2000.
- [12] D. Hart, 'Risk Management, Clinical Guidelines, Clinical Pathways and Health Law', *European Journal of Health Law*, **10**, 219–222, (2003).
- [13] D. Isern and A. Moreno, 'Agent-Based Careflow Usign CPGs', in *Recent Advances in Artificial Intelligence Research and Development*, eds., J. Vitri, P. Radeva, and I. Aguil, volume 113 of *Frontiers in Artificial Intelligence and Applications*, pp. 11–18, Barcelona, Catalonia, (2004). IOS Press.
- [14] D. Isern and A. Moreno, 'Distributed guideline-based health care system', in *4th International Conference on Intelligent Systems Design and Applications (ISDA-2004)*, pp. 145–150, Budapest, Hungary, (2004). IEEE Press.
- [15] D. Isern, A. Valls, and A. Moreno, 'Using Aggregation operators to personalize agent-based medical services', in *10th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, KES2006*, LNAI, p. to appear, Bournemouth, UK, (2006). Springer Verlag.
- [16] A. Kumar, S. Quaglini, M. Stefanelli, P. Ciccarese, E. Caffi, and L. Boiocchi, 'A framework for representing and executing a clinical practice guideline for the management of high blood pressure in pregnancy', *Technology and Health Care*, **10**, 517–519, (2002).
- [17] O. Lassila and R.R. Swick, 'Resource Description Framework (RDF): Model and Syntax Specification. W3C Recommendation', Technical Report REC-rdf-syntax-19990222, W3C, (1999).
- [18] L. M. Lau and S. Shakib, 'Towards Data Interoperability: Practical Issues in Terminology Implementation and Mapping', in *Thirteenth National Health Informatics Conference, HIC 2005*, pp. 208–213, Melbourne, Australia, (2005). Health Informatics Society of Australia.
- [19] A. Moreno, D. Isern, and D. Sánchez, 'Provision of agent-based health care services', *AI Communications*, **16**, 167–178, (2003).
- [20] A. Moreno, D. Snchez, and D. Isern, 'Security Measures in a Medical Multi-Agent System', in *Artificial Intelligence Research and Development*, eds., I. Aguil, Ll. Valverde, and Escrig M.T., volume 100 of *Frontiers in Artificial Intelligence and Applications*, 244–255, IOS Press, Amsterdam, The Netherlands, (2003).
- [21] J. L. Nealon and A. Moreno, 'Agent-Based Applications in Health Care', in *Applications of Software Agent Technology in the Health Care Domain*, eds., J. L. Nealon and A. Moreno, Whitestein Series in Software Agent Technologies, 3–18, Birkhuser Verlag, Basel, Switzerland, (2003).
- [22] M. Peleg, S. Tu, J. Bury, P. Ciccarese, J. Fox, R. A. Greenes, R. Hall, P. D. Johnson, N. Jones, A. Kumar, S. Miksch, S. Quaglini, A. Seyfang, E. H. Shortliffe, and M. Stefanelli, 'Comparing Computer-Interpretable Guideline Models: A Case-Study Approach', *Journal of the American Medical Informatics Association*, **10**, 52–68, (2003).
- [23] S. G. Priori, W. Klein, and J.P. Bassand, 'Medical Practice Guidelines: Separating science from economics', *Eur Heart J*, **24**(21), 1962–1964, (2003).
- [24] Agree project consortium, 'Development and validation of an international appraisal instrument for assessing the quality of clinical practice guidelines: the AGREE project', *Quality and Safety in Health Care*, **12**(1), 18–23, (2003).
- [25] S. Quaglini, P. Ciccarese, G. Micieli, and A. Cavallini, 'Non-Compliance with Guidelines: Motivations and Consequences in a case study', in *Computer-based Support for Clinical Guidelines and Protocols. Proceedings of the Symposium on Computerized Guidelines and Protocols, CGP 2004*, eds., K. Kaiser, S. Miksch, and S.W. Tu, volume 101 of *Studies in Health Technology and Informatics*, pp. 75–87, Viena, AU, (2004). IOS Press.
- [26] S. Quaglini, M. Stefanelli, G. Lanzola, V. Caporusso, and S. Panzarasa, 'Flexible guideline-based patient careflow systems', *Artificial Intelligence in Medicine*, **22**, 65–80, (April 2001).
- [27] D. Riaño, 'Ordered Time-Independent CIG Learning', in *V International Symposium on Biological and Medical Data Analysis (ISBMDA-2004)*, eds., Jos M. Barreiro, Fernando Martin-Sanchez, and et al. Vc-

- tor Maojo, volume 3337 of *LNCS*, pp. 117–128, Barcelona, (2004). Springer Verlag.
- [28] Y. Shahar and al., ‘A Framework for a Distributed, Hybrid, Multiple-Ontology Clinical-Guideline Library and Automated Guideline-Support Tools’, *Journal of Biomedical Informatics*, **37**, 325–344, (2004).
- [29] 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*, **36**, 193–272, (2006).
- [30] P. Terenziani, C. Carlini, and S. Montani, ‘Towards a comprehensive treatment of temporal constraints in clinical guidelines’, in *Ninth International Symposium on Temporal Representation and Reasoning (TIME 2002)*, pp. 20–27, Manchester, UK, (2002). IEEE Press.
- [31] P. Terenziani, S. Montani, A. Bottrighi, G. Molino, and M. Torchio, ‘Clinical Guidelines Adaptation: Managing Authoring and Versioning Issues’, in *10th Conference on Artificial Intelligence in Medicine, AIME 2005*, eds., S. Miksch, J. Hunter, and E. Keravnou, volume 3581 of *LNAI*, pp. 151–155, Aberdeen, Scotland, (2005). Springer Verlag.
- [32] S. W. Tu and al., ‘Modeling Guidelines for Integration into Clinical Workflow’, in *Medinfo 2004*, eds., M. Fieschi, E. Coiera, and Y. J. Li, volume 107 of *Studies in Health Technology and Informatics*, pp. 174–178, San Francisco, US, (2004). IOS Press.
- [33] S. W. Tu and al., ‘Use of Declarative Statements in Creating and Maintaining Computer-Interpretable Knowledge Bases for Guideline-Based Care’, Technical report, Stanford Medical Informatics (SMI), (2006).
- [34] S.W. Tu and M.A. Musen, ‘Modeling Data and Knowledge in the EON Guideline Architecture’, in *MedInfo 2001*, eds., V. Patel, R. Rogers, and R. Haux, volume 84 of *Studies in Health Technology and Informatics*, pp. 280–284, London, UK, (2001). IOS Press.
- [35] 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*, **37**, 305–318, (2004).
- [36] O. Young and Y. Shahar, ‘The Spock System: Developing a Runtime Application Engine for Hybrid-Asbru Guidelines’, in *10th Conference on Artificial Intelligence in Medicine, AIME 2005*, eds., S. Miksch, J. Hunter, and E. Keravnou, volume 3581 of *LNAI*, pp. 166–170, Aberdeen, Scotland, (2005). Springer Verlag.