Proteus – A Model for Clinical Protocols Created from Knowledge Components

Hemant Shah
National Library of Medicine, Bethesda, MD, USA
h.shah@computer.org

Abstract

We describe a model – Proteus, for clinical protocols created with clinical knowledge represented as components. By applying principles of distributed computing the knowledge can be managed by experts at other locations while it is available to the clinicians as executable elements for decision support. Editability in the hands of users is essential for the success of any decision support system. A component approach for knowledge along with a notation system for representing the components allows editability of the protocols by the users. The knowledge components (KCs) also serve as components for an electronic medical record (EMR). Since KCs represent activities within the clinical process, they provide easy way to link components of other clinical or non-clinical processes that are associated or dependent. This provides for extensibility and allows conceiving of integrated information systems that can deal with diverse aspects of healthcare in the Proteus approach.

1. Introduction

Many approaches and models have been proposed to offer clinical guideline based support to clinicians [1-6]. However, creation, maintenance and customization of the knowledge comprising protocols by the users has not been emphasized enough in these approaches. We describe a model in which editable protocols are created from distributed, reusable and executable knowledge components. Besides editability our approach offers several other advantages. A significant advantage is possibility of maintenance of the knowledge by experts situated at different locations, while the users, with no effort get the benefit of the executable up-to-date knowledge. Also, the component approach has been utilized to conceive of the inference tool as a pluggable entity allowing tools from diverse locations or technologies to be used.

2. Editability – an essential feature

Editability of protocols is essential for following reasons. **Site-Specificity:** Knowledge available in sources like journals and textbooks or even in the electronic protocols is never directly applicable; a great part of the information required for making decisions comes not from standard medical knowledge but local factors. These factors include population characteristics, disease patterns, social and legal issues, and medical setup and skills. Only editability in the hands of users can render a protocol pragmatic to a location. **Unpredictability:** Editability in the hands of the clinician while the protocol is running would allow dealing with the ad hoc nature of clinical process. **Unique Standards for each location:** The ability to modify the standard guidelines allows the institutions to declare its own best practices. Additionally, each provider's performance can be evaluated in light of its own standards rather than those created elsewhere. **Variations in clinical problems and rapid evolution of knowledge:** The many variations in diseases and the rapid changes in medical concepts preclude creating applications that have such knowledge embedded in them. Clinical protocols have to be treated as being editable and separate from applications in which they execute. This makes the same application useful for all variations and also when concepts change. Musen et al [[7]] and Wiener [[8]] have also discussed benefits of editing decision-support systems by clinicians.

3. Other features essential for executable protocols

To be successful, executable protocols, like any other decision support approach will need to be well integrated with healthcare information systems [9-12]. Also, protocol-based systems will have to provide the means for updating the protocols to reflect the current understanding of medical concepts as they evolve.

4. Proteus – PROTocols Editable by USers

The key requirements for a protocol system aiming to provide modifiability by the users are **(a)** The underlying knowledge representation scheme should be well understood by the user, as well as the medical semantics of the protocols themselves. **(b)** The constituent elements of the protocols should be relatively independent so that changing one does not disrupt others. In our model of executable protocols, Proteus, these requirements were met by expressing protocols as being composed of knowledge elements in form of components – the *knowledge component* (KC). Each type of KC has an unambiguous clinical meaning and a corresponding graphical icon, making it comprehensible to the users and amenable to easy modifications.

The KCs, which represent clinical activities, contain **(a)** data elements or other KCs. **(b)** *Inference Tools* that are responsible for any transforms or activities within the KC that are intelligence driven, and **(c)** *Activity Links* that define the workflow relationship between the contained KCs. Protocols are created by assembling the KCs. The Proteus notation system has elements that correspond well with the underlying entities, and is expressive enough to represent complex clinical activities.

Since the clinical activity, workflow and inferencing concerns are neatly segregated, change in one element does not affect others and it is easy to rearrange the elements. The KCs are distributable components allowing these to be managed by experts from remote locations so that the users may get benefit of current knowledge automatically. As KCs contain the data elements they also serve as segments of an EMR, an EMR that is distributed, process-oriented and rich in content since besides data it also stores the interpretations and decisions with their basis and the contexts in which these were created. KC being a decision-support element as well as an EMR component, there is an automatic 2-way linkage between the two, without requiring the data to be entered separately for decision making tools or without losing the data that is entered for getting the decisions. Since Proteus protocols are a representation of the clinical process, the KCs within them provide convenient points for anchoring information elements from many other processes in healthcare, dependent on the clinical process. This allows Proteus protocols to be conceived as a core for comprehensive healthcare information systems.

5. Knowledge Components

KCs are of two types – The *transaction KC* and the *process KC*. The transaction KC represents clinical event or action and contains the primitive data elements that describe the underlying event or action. Transaction KC is where the Proteus based system gathers the data from the clinical world. The process KC represents a clinical process and contains nested elements, which may be transaction KCs or other process KCs. It also has activity links to represent which activity follows which. The protocol itself is a process KC but at the highest level. Figure 1 shows a process KC with nested transaction KCs.

Being distributed components, the KCs can only be accessed by their interfaces. The KCs are not just components in computing sense but also at the clinical semantic level where the clinician has to interact with it. Therefore, each KC also has a *semantic interface*, where the KC exposes its name, value (abstraction value, see section "**Inference tools**") and the possible values that it can possess. A KC has to stay faithful to this interface even if internally it undergoes radical modifications. Other elements external to a KC deal only with this interface, and therefore are not concerned with if it has been changed. Furthermore, the interface also conceals internal complexity behind this interface – internally a KC may contain numerous nested elements but the entities external to it treat it only as a simple data element with a single value. The KCs also do not have any knowledge of elements external to it. Thus, we have applied the principles of information encapsulation and distributed components to the KCs at a semantic level to facilitate editability by users. Additionally, treating protocol elements as distributed components means that experts, located even at different locations, could maintain and enhance them allowing the users to get the benefit of the state-of-the-art clinical knowledge. The components themselves may be located at a central location, at the experts' organizations, or at the user facilities.

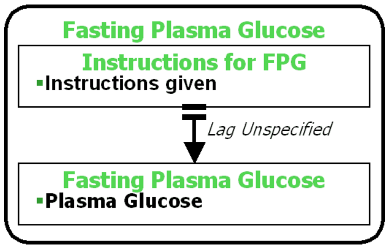

Figure 1. A process KC in Proteus notation with two nested transaction KCs.

6. Activity Links

The activities represented by KCs may be linked together by using *activity links*, which denote activity transitions. The links serve to put the different activities in context of workflow. There are several types of links based upon how they are triggered and what action do they result in. Figure 2 shows the commonly used types of activity links in Proteus notation format. An arrowhead at the target-end represents activation of the KC to which it is connected. All the links shown here are of the activation type. The source-end of the arrow represents how the activity transition is triggered. Inferential link has a filled square at the trigger-end and denotes activation via an inferential mechanism. Sequential link has a double bar at its trigger end and conveys that activity is triggered simply after the source activity is over. In case of Synchronous (concurrent) link, the activity at the source and destination are triggered simultaneously. The links may optionally have text-labels describing the inference that leads to their triggering, for any time lag between triggering and its resultant action, or in case of a reiterative situation how many times and at what frequency will it be triggered.

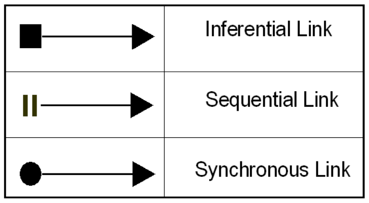

Figure 2. Proteus notation for the three commonest types of links used to represent activity transitions.

7. Inference tools

KCs use two types of inference tools. All KCs have at least one inference tool – *abstraction inference tool* (AbIT) that creates an *abstraction* for them. *Abstraction* is a value for the KC, aggregately representing the value of all elements (primitive data elements or other KCs) within it. For instance, a KC – "Symptoms of Diabetes" would have abstraction value *true* if its data-elements "Polydipsia", "Polyuria" and "Unexplained weight loss" all had values *true*. Each time any data changes within a KC the AbIT is triggered and it changes the abstraction for the KC. It is by means of abstraction that the KC presents a single value to entities external to it, regardless of how many items it contains internally. Abstraction also provides a mapping of activities to states allowing changes in abstractions of the KCs to be treated as state transitions.

Process KCs may additionally have an *action inference tool* (AcIT) to decide which activity is to be triggered next within them. The task of the AcIT is to decide if one or more inferential links emanating form a contained KC are to be triggered or not. The other types of links do not require inferencing to decide their triggering as they have simple logic of their own.

The inference tools in the Proteus protocol model are specified only as an interface, which means that any inference tool that complies with the interface may be deployed. This allows use of tools from diverse technologies like artificial neural networks, rule based systems, etc., appropriate for the task. Indeed, any technology can be used if couched in the interface for an inference tool. The tool may even be located remotely. Human expertise could also be deployed as inference tool, providing a framework for distributed collaborative decision-making approach. This also allows for the patient's role in decision-making. Interface based access also means that a tool may be swapped for another easily. The inference tools do not have any notational representation.

8. Layers for extensibility

Since each KC represents something distinctly identifiable within the clinical process, they are convenient anchor points for components of other (non-clinical) processes in healthcare. For example, though the clinician may be concerned with the information that would help make decisions about procedures and the data to be collected from them, the healthcare manager would be interested in providing the resources and logistics to facilitate and optimize the process, and to monitor the activity. Similarly, other healthcare personnel may be linked with the core (clinical) process. Each such process can be defined as a layer. The core layer keeps other layers, such as those for the administrator, researcher, account section or others informed about events within it. The layers feature allows the Proteus approach to be the basis of a comprehensive and integrated healthcare information system. New layers, even ad hoc ones can be created, allowing unlimited extensibility. Layers may be created to support and enhance the clinical process also. For instance, a layer may be created to allow HL7 based messaging for each KC.

9. An overview of Proteus based system

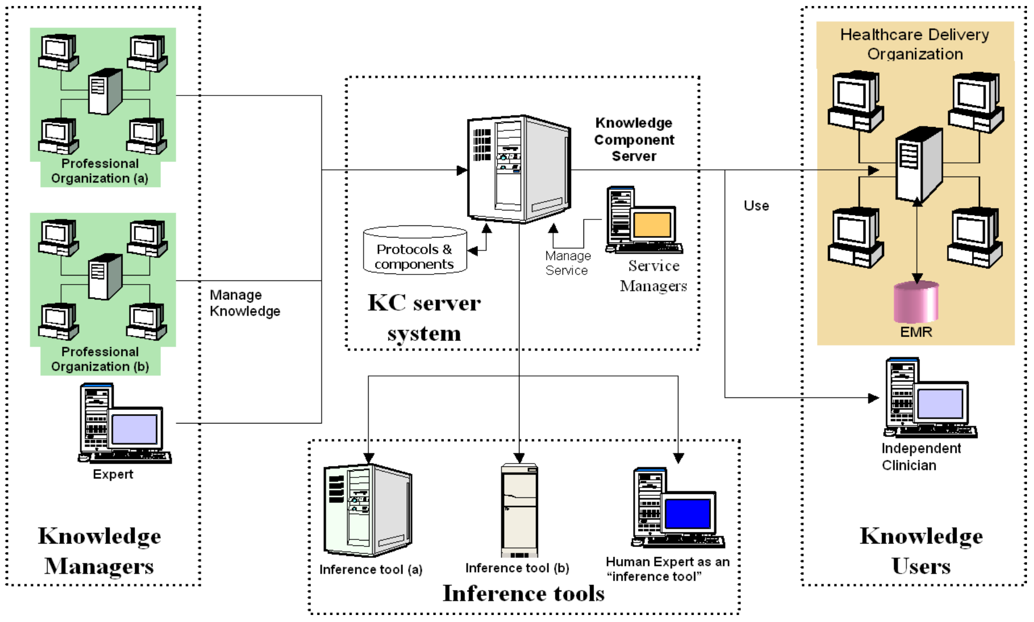

Figure 3. A Proteus based system

The core facility in a Proteus-based clinical decision-support system is the **knowledge component server** (see Figure 3) providing access to the KCs including the protocols. These may be hosted in the server facility or may be made available via a naming and directory service. The inference tools for the KCs may be located at other secure sites. Human experts could also be designated as inference tools for certain KCs. Professional organizations and individual experts (**knowledge managers**) would be authorized to manage the knowledge on the KC server from their respective locations. The knowledge managers use compliant client tools for creating and updating the KCs some of which may even be hosted at their own sites. The users (**knowledge users**) access the protocols and KCs from the KC server, also using the client software which also allow the users to see the patient data, thus serving as the front end of the electronic medical record. The EMR created, based on the templates that KCs provide is stored at the providers' facility. The users may also store at their own location the KCs they have modified after accessing them from the knowledge server, or the ones they create themselves.

10. Execution and inferencing

When a protocol is executed, the system identifies the first KC to be launched; if it is a transaction KC it is shown to the user as a dialog-box for data-entry (or interacts with other designated data-entry agent for automated data-entry); if it is a process KC, the system looks within it, at progressively deeper nesting levels, until a KC is found that is transaction KC. Any time execution reaches a transaction KC it retrieves the data from the external world. Once data is entered, the AbIT of the transaction KC creates an abstraction for it, since any change in the contents can potentially change the value of the container. If the value of the transaction KC is changed, it triggers the AbIT of its own container process KC. This process is repeated for all container process KCs till abstraction of the top-level process KC (protocol) is also changed. This process, known as *reverse abstraction cascade*, ensures that the implication of any data change is reflected at all levels of granularity, including the global one for the patient. After the abstraction process is done the container process KC of the transaction KC last launched decides which activity is to be launched next. If the activities to be triggered are sequential they are directly launched, however if the KC which finished executing last is the trigger for any inferential activity links, the AcIT of its container decides if any and which of the inferential links available have to be triggered. The data collected in the process, with the abstractions and actions are stored as EMR for the patient.

11. A prototype system

At NLM, we are developing a prototype system to evaluate the Proteus approach. In an Enterprise JavaBeans compliant architecture, the KCs are entity beans and inference tools are session beans. We are using Borland® AppServer 4.5™ for deployment of these beans. The client tool, **Protean 2.0** serves as a standalone application also. We have been testing the tools using several sample protocols. Protean reads the protocols, displays them in Proteus notation, executes them, and allows their editing. To demonstrate the power of interface based access of inference tools we use as inference tools, algorithmic tools as Java classes, and two rule engines, between which we can switch easily. We are providing Jess – a Java based rule-engine as the default inference tool. Protean has a rule editor that allows rapid development of Jess rules and their maintenance, one KC at a time.

References

[1] Musen MA, Tu SW, Das AK and Shahar Y, "EON: a component-based approach to automation of protocol-directed therapy", J Am Med Inform Assoc., 1996 Nov-Dec, 3 (6): 367-88.
[2] Ohno-Machado L, Gennari JH, Murphy S et al, "The guideline interchange format: a model for representing guidelines", J Am Med Inform Assoc., 1998 Jul-Aug, 5 (4):357-72.
[3] Stoufflet PE, Ohno Machado L, Deibel SRA et al, "GEODE-CM: a state-transition framework for clinical management", Proc 20th Annu Symp Comput Appl Med Care, 1996, 924.
[4] Barnes M and Barnett GO, "An architecture for a distributed guideline server", Proc Annu Symp Comput Appl Med Care., 1995, 233-7.
[5] Quaglini S, Dazzi L, Gatti L et al., "Supporting tools for guideline development and dissemination", Artif Intell Med., 1998 Sep-Oct, 14(1-2):119-37.
[6] Shahar Y, Miksch S and Johnson P, "The Asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines", Artif Intell Med., 1998 Sep-Oct, 14 (1-2):29-51.
[7] Musen MA, Fagan LM, Combs DM and Shortliffe EH, Use of a domain model to drive an interactive knowledge-editing tool, International Journal of Man–Machine Studies, 1987, 26:105–121.
[8] Wiener F, "SMR (simulating medical reasoning): an expert shell for non-AI experts", Comput Methods Programs Biomed. 1988 Jan-Feb,26 (1):19-31.
[9] Safran C, Rind DM, Sands DZ, Davis RB, "Development of a knowledge-based electronic patient record", MD Comput., 1996 Jan-Feb,13 (1):46-54, 63.
[10] Weed LL, "New connections between medical knowledge and patient care", BMJ, 1997 (26 July), 315:231-235.
[11] Rector AL, Nowlan WA and Kay S, "Foundations for an electronic medical record", Methods Inf Med., 1991 Aug, 30(3):179-86
[12] Owens DK, "Use of medical informatics to implement and develop clinical practice guidelines", West J Med., 1998 Mar, 168(3):166-75