Guideline Knowledge Representation Model (GLIKREM)

David Buchtela, Jan Peleška, Arnošt Veselý, Jana Zvárová, Miroslav Zvolský
Centre of Biomedical Informatics, Institute of Computer Science AS CR, Prague, Czech Republic

Summary: The guideline knowledge representation system (GLIKREM) is based on a GLIF model which was published in a GLIF3.5 specification. GLIKREM contains some changes and extensions of the definition and implementation of the original GLIF model. The aim of this article is to give a description of GLIKREM, its construction, its implementation in XML, a realization of the data interface and use of the result model.

Keywords: knowledge representation, GLIF model, guidelines

Introduction

Medical guidelines are used for clinician decision support. They are intended to improve the quality of patient care and reduce costs. Unfortunately, finding information contained in conventional (free text form) guidelines may be difficult. A prerequisite for developing decision support systems that use guidelines is creating computer interpretable representations of the medical knowledge contained in medical guidelines. A number of groups are actively developing computer interpretable guideline representation languages for this purpose. The Arden Syntax [1] is perhaps the best known language for representing medical knowledge in decision support systems. It is a rule based formalism developed for encoding individual clinical rules as Medical Logic Modules. Other approaches share a hierarchical decomposition of guidelines into networks of component tasks that unfold over time [2]. All of the following modelling methods can combine guideline steps in directed cyclical graphs.
Asbru is being collaboratively developed at Ben Gurion University and the Vienna University of Technology [3]. It is a time oriented, intention based, skeletal plan specification language that is used to represent clinical protocols.

EON was developed at Stanford University and provides a suite of models and software components for creating guideline-based applications [4]. EON uses a task based approach to define decision support services that can be implemented using alternative techniques [5]. They use the Protégé-2000 environment to build a patient-data information model, a medical-specialty model, and a guideline model that formalizes the knowledge needed to generate recommendations regarding clinical decisions and actions.

GUIDE is part of a guideline modelling and execution framework developed at the University of Pavia [6]. It supports integrating modelled guidelines into organizational workflows, using decision analytical models such as decision trees and influence diagrams, and simulating guideline implementation in terms of Petri nets.

PRODIGY was developed at the University of Newcastle upon Tyne [7]. The PRODIGY project's aim is to produce the simplest, most readily comprehensible model necessary to represent the class of guidelines. Teams of clinicians have used Protégé's knowledge engineering environment [8] to encode three complex chronic disease management guidelines.
PROforma was developed at the Advanced Computation Laboratory of Cancer Research [9] in the United Kingdom. It combines logic programming and object oriented modelling and is formally grounded in the R2L Language. PROforma supports four tasks: actions, compound plans, decisions, and enquiries. All tasks share attributes describing goals, control flow, preconditions, and post conditions.

The Guideline Interchange Format version 3 (GLIF) has been collaboratively developed by groups at Columbia, Stanford and Harvard Universities (the InterMed Collaboratory). The most recent GLIF specification is the GLIF3.5 specification [10,11,12]. Its expression language was originally based on the Arden Syntax. GLIF used the Guideline Expression Language (GEL) to specify criteria and expressions. GLIF now uses GELLO, an extensible object oriented expression language that supports a superset of the functions supported by GEL [13].

GLIF was selected as the model for representing the Computerized Clinical Guidelines in SAPHIRE too [14]. The SAPHIRE system continuously monitors the patients through dedicated agents and supports the healthcare professionals through an intelligent decision support system.

The aim of this article is to describe the design of a guideline knowledge representation model (GLIKREM). GLIKREM is based on a GLIF model. GLIKREM contains some changes in and extensions of the definition and implementation of the original GLIF model. This article describes GLIKREM, its construction, its implementation in XML (eXtensible Markup Language), a realization of the data interface and the use of the model. GLIKREM is used in the Medical Knowledge Representation System (MEKRES) which is developed now [15].

Guideline knowledge representation system

The whole process of the construction of a knowledge model (GLIKREM) from the free text of guidelines, its representation in XML and the use of the model is illustrated in figure 1 [16].

In the stage of model construction from textual guidelines it is important to find the logical and process structure of guidelines (the decision algorithm), all fundamental

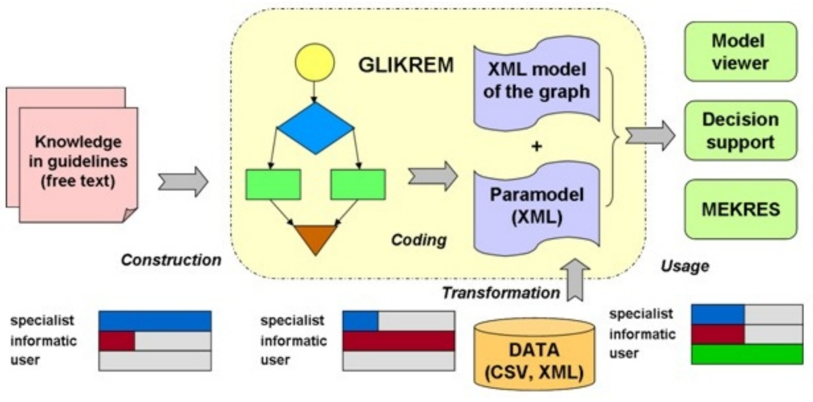

Figure 1. Process of construction, coding and use of GLIKREM. Source [1].

parameters and their interrelationships [17].

In the stage of the model coding the graphic model of guidelines is coded into XML. Besides this a list of basic and derived parameters of the model (paramodel) is created.

The paramodel is used as an interface between the model and real data [18]. In the stage of model construction from textual guidelines it is important to find the logical and process structure of guidelines (the decision algorithm), all fundamental parameters and their interrelationships [17].

In the stage of the model coding the graphic model of guidelines is coded into XML. Besides this a list of basic and derived parameters of the model (paramodel) is created. The paramodel is used as an interface between the model and real data [18].

Graphic model

Main parts (steps) of graph

The model of textual guidelines created in the construction stage is an oriented graph (see fig. 2) which is composed of five main parts (steps):

- **Action step** – specify clinical actions that are to be performed. It can be an application of some therapy, carrying out some examination or measurement etc. Action steps also may refer to sub-guidelines (subgraph), which provide details of the action.
- **Decision steps** are used for conditional branching. These steps are used when branching is determined by evaluation of logical criteria based on the value of data items. If the decision cannot be made automatically the user can select himself one of the alternative next steps.
- **Branch** and **synchronization steps** enable concurrence in the model. Guideline steps that follow a branch step can be performed concurrently. Branches have a root in the branch step and eventually converge in a synchronization step. In this step all branches are synchronized after evaluation of a synchronizing condition.

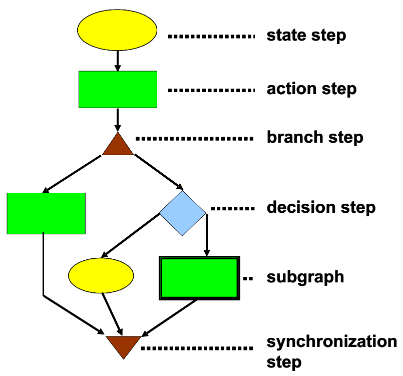

Figure 2. Parts (steps) of the graphic model.

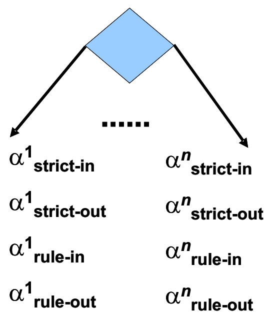
......
$\alpha^1_{\text{strict-in}}$
$\alpha^1_{\text{strict-out}}$
$\alpha^1_{\text{rule-in}}$
$\alpha^1_{\text{rule-out}}$
$\alpha^n_{\text{strict-in}}$
$\alpha^n_{\text{strict-out}}$
$\alpha^n_{\text{rule-in}}$
$\alpha^n_{\text{rule-out}}$

Figure 3. Decision criteria in a decision step.

- **State step** characterizes an object state that has to be observed after execution of the previous step or a state that should exist at the beginning of the model.

Decision criteria

Each decision step specifies four criteria for each decision option α^1 ... α^n (see fig. 3). The subsequent flow through the model is automatically or manually chosen on the basis of the evaluation of these criteria.

- *Strict-in* – if a *strict-in* is true then the control flows to the guideline step that is specified by that decision option's destination.
- *Strict-out* – if a *strict-out* is true then the decision option's destination is forbidden. The strict-out can be but do not have to be opposite of strict-in.
- *Rule-in* – if a *rule-in* is true then it is only recommended to flow to the guideline step that is specified by that decision option's destination. The user should select himself one of the next steps with positive rule-in.
- *Rule-out* – if a *rule-out* is true the decision option's destination is not recommended but it is not forbidden. The user should not select one of the next steps with positive rule-out.

The *strict-out* criterion is evaluated at first. If the *strict-out* criterion (of some option) is evaluated as true the rest of the criteria (of this option) is not evaluated. This option is forbidden. In the opposite case the *strict-in* criterion is evaluated. If the both of *strict-in* and *strict-out* criteria are false, the *rule-in* and *rule-out* criteria are evaluated. The ranking of *rule-ins* and *rule-outs* (of all option's critearia) is left to the users who may use their clinical judgement or may develop their own ranking schemes.

Criteria evaluation

When evaluating the criteria (*strict-in, strict-out, rule-in, rule-out*), it often happens that input parameter values are not known. Therefore the criteria are evaluated in three-value (or multi-value) logic. The logic formulas contain variables (parameters of the paramodel) and logic or relational operands and can take the values:

- **true** – the result of the formula evaluation is true – it is represented by the value 1.

- **false** – the result of the formula evaluation is false – it is represented by the value 0.
- **unknown** – when the formula cannot be evaluated – it is represented by the value ½.

Each logic formula should be expressed in the disjunctive or conjunctive normal form that contains only the logical operations of negation, conjunction (logical multiply) or disjunction (logical sum). This form is equivalent to the original formula.

The definition of the operations of negation, conjunction and disjunction in three-value logic are as follows:

- **negation:** $\neg p = 1 - p$
- **conjunction:** $(p_1 \wedge p_2 \wedge \ldots \wedge p_n) = \min(p_1, p_2, \ldots, p_n)$
- **disjunction:** $(p_1 \vee p_2 \vee \ldots \vee p_n) = \max(p_1, p_2, \ldots, p_n)$

A well-designed model has to fulfil several rules for each decision step:

- The automatic flow to the decision option's destination is possible if its *strict-in* criterion is true and its *strict-out* criterion is false.
- If the first rule does not happen the user must chose some destination manually. He (she) can use only the destination whose *strict-out* criterion is false.
- The recommended and not recommended options are offered to the user after the evaluation of *rule-in* and *rule-out* criteria.
- The *strict-in* and *strict-out* criteria must have definite values (true or false). If some *strict-in* or *strict-out* criterion is evaluated as unknown, the user will have to add the missing data.
- Each decision step must be consistent i.e. at the most one of the *strict-in* criteria (of all options) can be true and both of *strict-in* and *strict-out* criteria are not true in the same option.
- Criteria can be dependent on several different parameters. The evaluation of the decision step must be correct (at least one of the *strict-in* or *rule-in* is true in some option and *strict-out* is false in this option) for all of possible values of each parameter.

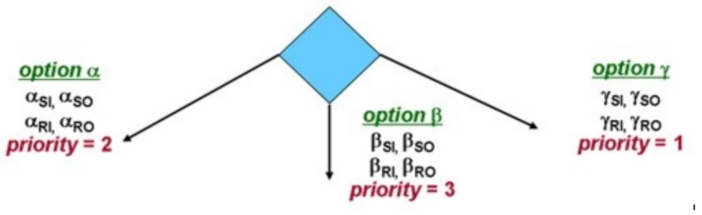

Figure 4. Option priority in the decision step.

Priority in the decision

If these rules are thoroughly applied the user can insert missing data when it is necessary. If a strict-in criterion of some option (destination) is true the evaluation of criteria of other options is not needed. The amount of necessary essential data is dependent on order of single option evaluations. Therefore it is necessary to set an order of evaluation i.e. to set the priority of decision options. The priority of each option is chosen by a specialist (see fig.4).

In the example (see fig. 4) the criteria of the option γ is evaluated first then the criteria of the option α and finally the criteria of the option β. If strict-in criterion of the option γ is true the options α and β will be not evaluated.

Synchronization conditions

In modelling parallel branches (with branch and synchronization steps) it is necessary to specify which branch is necessary and which one is optional. That is why the synchronization conditions are set for each synchronization step (see fig.5).

The synchronization condition has to be expressed in the disjunctive or conjunctive normal form. The onfly parameters of the paramodel are used as variables in the synchronization condition. Values of onfly parameters are:

- value 1 – the user goes through the branch which is represented by this onfly parameter,
- value 0 – the user do not go through this branch.

A value of the onfly parameter is set by a "shadow" operation (see XML representation of model) in the last step of the branch.

In the example (see fig. 5) the synchronization condition $(\alpha_1 \vee \alpha_2) \wedge \alpha_3$ will be true if

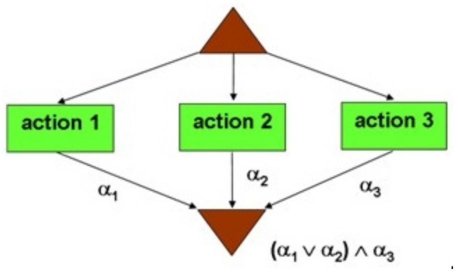

Figure 5. Synchronization condition.

the user goes through branch α_3 and goes through one of the branches α_1 or α_2.

Representation of the graphic model in XML

We specified our own XML scheme for representation of graphic model and developed a graphical editor for construction of the graphical model and its translation into XML. The whole model consists of a sequence of XML elements which represent steps of the graphic model. The XML syntax of the model contains elements for description of attributes of model steps, elements for graphical symbols of steps and elements for the decision support (see fig. 6).

The description of elements is as follows:

- **glikrem** – the root element of the whole model (GLIKREM),
- **head** – the definition of head – see the paragraph Model extension for using in MEKRES,
- **graph** – the root element of the graphic model contains elements step,
- **step** – the attributes of one step, it contains elements:
  name – the unique identification of the step,
  type – the type of the step, the type can be: *action, case, branch, synchronization, state, subgraph,*
  caption – the caption of the graphical symbol of the step,

- **text** – the description of the step,
  x – the x-coordinate of the graphical symbol,
  y – the y-coordinate of the graphical symbol,
  width – the width of the graphical symbol (in pixels),
  height – the height of the graphical symbol (in pixels),
  focus – the highlighting of the step for visualization of a path in the graph, possible values are:
  not = not highlighted,
  auto = automatically highlighted,
  user = highlighted by the user simulation,
  status – the location of the step in the model, possible values are:
  start = the first step (the graphical model starts with this step),
  end = the last step (usually the final patient's state),
  in = the internal step,
- **operation** – the list of "shadow" operations of the step (see fig. 7), each operation (element ops) contains elements:
  otype – the type of operation, possible values are:
  insert = the insertion of the parameter value,
  get = the getting of the parameter value,
  put = the putting of the parameter value,
  open = the opening of some file (e.g. subgraph),
  oparam – the name of the parameter or the file,
  ovalue – the value onfly for *put* operation,
- **next** – the list of follow branches (options), each option (element option) contains elements (see fig. 8):
  nname – the identification of the option,
  nstep – the identification of the target step,
  ncaption – the caption of the option,
  npriority – the priority of the option, the default value is 1,
  nstrictin – *strict-in* criterion,
  nstrictout – *strict-out* criterion,
  nrulein – *rule-in* criterion,
  nruleout – *rule-out* criterion,
  nnote – the note of the option,
  nline – the description of line to the target step in a form of the sequence of points (elements point):

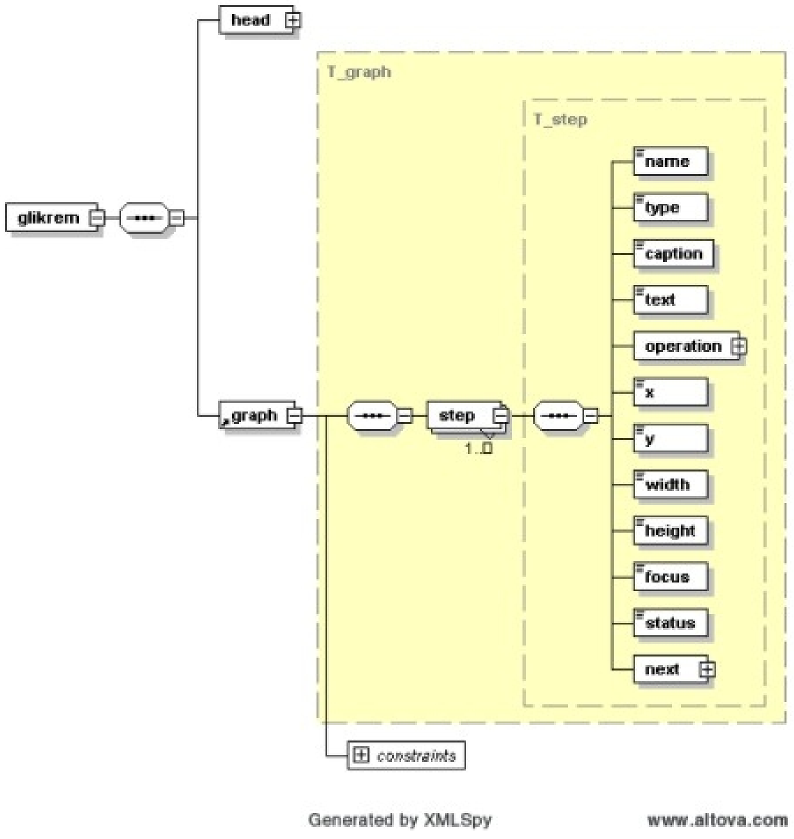

Figure 6. XML scheme of the graphic model representation.

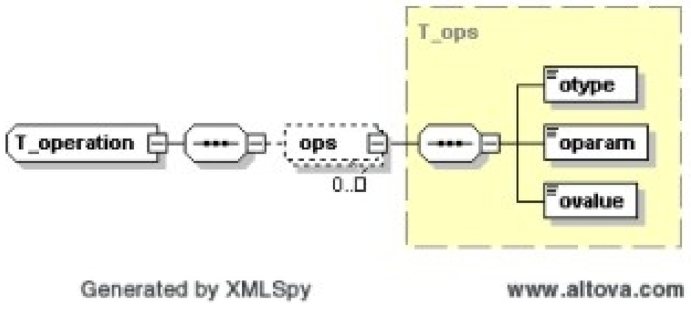

Figure 7. XML scheme of shadow operations.

px – the x-coordinate of the point,
py – the y-coordinate of the point.

Every type of decision condition is usually a logic formula with basic and derived parameters of the paramodel as variables in these formulas. The notation of the decision conditions (*strict-in, strict-out, rule-in, rule-out*) adheres the **XPath** syntax.

For example the definition of condition with parameters SYMPTOMS and FINDINGS: /params/param[pid="SYMPTOMS"]/pvalue **and** params/param[pid="FINDINGS"]/pvalue.

The notation of the synchronization conditions is analogical (in XPath) but we need onfly parameters as variables for these conditions.

For example the definition of synchronization condition with onfly parameters (branches) DG_S3 and DG_S4:/params/param[pid="DG_S3"]/pvalue **or** /params/param[pid="DG_S4"]/pvalue.

Model of parameters (paramodel)

The essential parameters for GLIKREM are often saved (in databases or clinical information systems) in a form which is not efficient for direct use in the knowledge model. The model of parameters (paramodel) is a better solution. The paramodel is constructed along with GLIKREM. The paramodel serves as a data interface between real data and GLIKREM. The values of parameters of the paramodel are derived from real data by data transformation (see the paragraph Data transformation). Only the parameter names of the paramodel are used in the decision conditions.

Representation of paramodel in XML

The paramodel is constructed by user in the graphic model editor and represented in the XML the same way as a graphic model. The whole paramodel contains elements which represent the respective parameter (see fig. 9). The parameters of the GLIKREM can be as follows:

- **basic** – the basic parameter is directly measurable or has a value which is found by an other method,
- **derived** – the parameter is derived from basic parameters by the application of a logical or arithmetical operation,
- **onfly** – the parameter for synchronization conditions which contains a logical value. The value is set by a "shadow" operation in the last step of the branch which is antecedent to the synchronization step.

The description of the paramodel elements is as follows:

- **paramodel** – the root element of the paramodel,
- **head** – the definition of the head,
- **params** – the root element of the parameters which contains elements param,
- **param** – the description of one parameter,
  pid – the unique identification of the parameter,
  ptype – the type of the parameter, possible values are: basic, derived and onfly,
  pname – the full name of the parameter,
  pdatype – the data type of the parameter (XML data type),
  pvalue – the value of the parameter,
  punits – the unit of the parameter,
  pdef – the definition of the derived parameter in Xpath,
  pquery – the definition of the database query (e.g. in SQL),
  pnote – the note of the parameter.

Transformation of input data

The real data of patients are stored in clinical information systems and databases of different types. It is usually possible to export data from these systems in the form of XML or CSV (Comma-Separated Values) files. From these files it is necessary to transform data into the paramodel (in XML). A simplified approach of the transformation you can see in figure 10. It is necessary to define the value-map that contains an input and a result. The input is the parameter and its value as exported by an XML-file or a CSV file and the result is a mapping to the parameter name and value as used in the paramodel. The definition of the value-map is in the XSLT (eXtensible Stylesheet Language Transformations) file form. For example the determination of the systolic blood pressure value as an average from measured values which are stored in the XML file. The input data are the values of the different systolic blood pressure measuring and the result is the average value (see fig. 11).

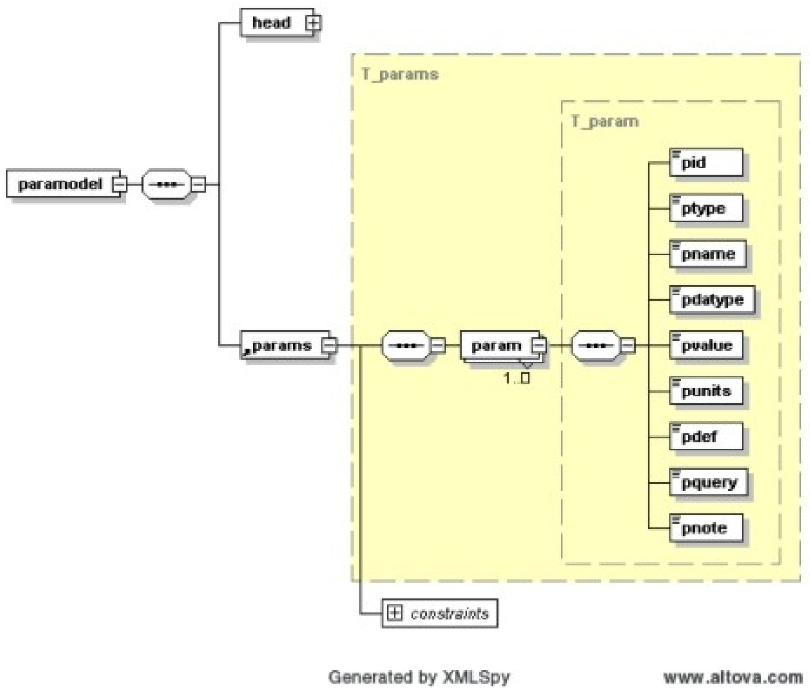

Figure 9. XML scheme of the paramodel.

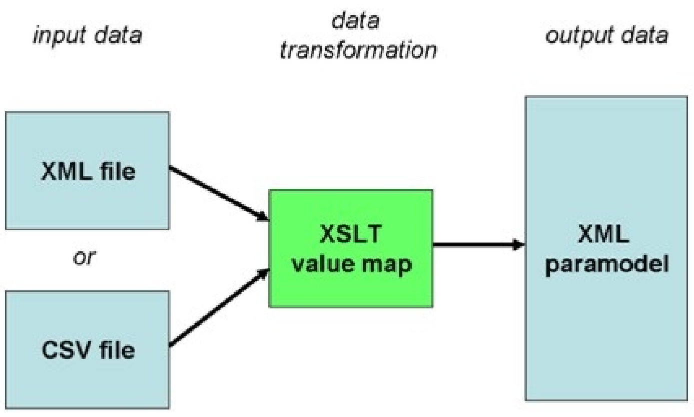

Figure 10. The process of the data transformation into the paramodel.

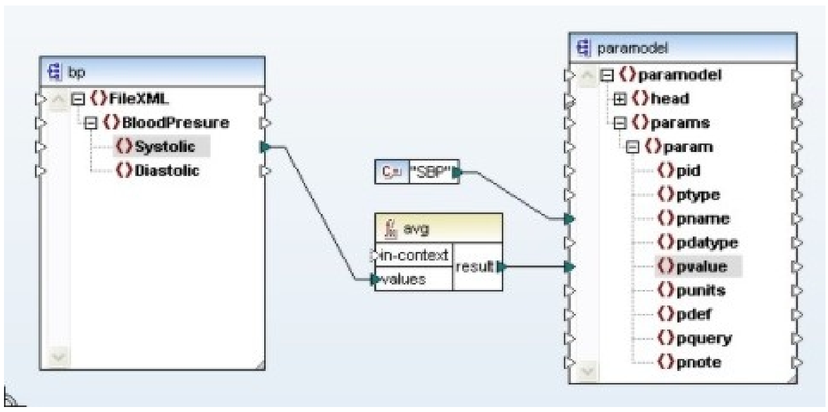

Figure 11. Transformation of systolic blood pressure value

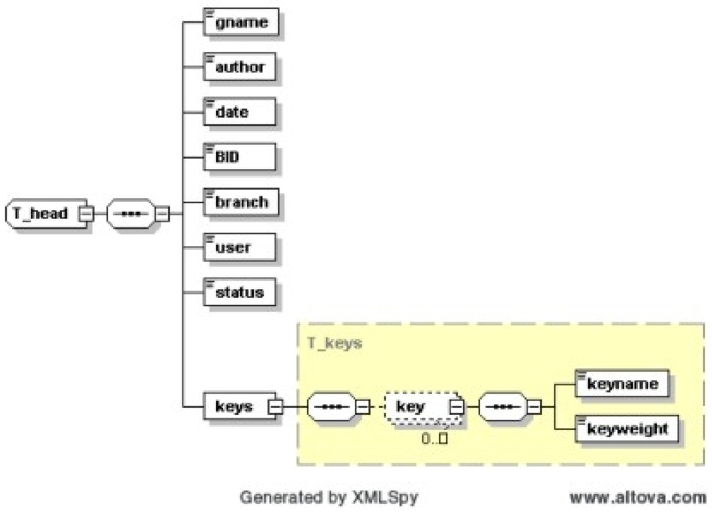

Figure 12. XML scheme of key atributes.

Model extension for using in MEKRES
The resulting GLIKREM is extended by key attributes for use in MEKRES (MEdical Knowledge REpresentation System). These key attributes are used in the selection algorithm of a relevant model and they are coded in XML and stored in the element head along with the graphic model (see fig. 12).

The description of the element head is as follows:

- **gname** – the name of knowledge model (GLIKREM),
- **author** – the author(s) of the model,
- **date** – the last update date,
- **BID** – the branch identification – e.g. International Classification of Diseases (ICD),
- **branch** – the branch described by the model,

user – the user of system to whom the model is primary determined, possible values are: *patient*, *GP*, *specialist*, *operator (of emergency rescue)*, *everybody*,

- **status** – the model validity, possible values are: *valid*, *draft*, *expired*,
- **keys** – the list of keys described by the model contains elements key, **keyname** – the name of the **key**, **keyweight** – the weight of the key – the model description rate of the key.

Selection algorithm of relevant model
The principle of the whole medical knowledge representation system (MEKRES) and the algorithm to select therelevant GLIKREM model is illustrated in figure 13. The select algorithm is described as follow:

- A set of all branches and keys (key attributes) which corresponds to participant state (his/her attributes) is determined for every specific participant (user, patient, physician, operator, ...) and his attributes.
  - e.g. branches "diabetes" and "hypertension" which corresponds to patient attributes "SBP", "DBP", "glycaemia".
- For each branch and key the set of knowledge models (GLIKREM) are chosen from a set of all GLIKREMs where each model contains attribute branch and key weight (element keyweight) is non zero.
  - e.g. models G_1 (hypertension) and G_2 (diabetes).
- For each chosen model a general aggregate operator (e.g. $R_g = \sum keyweight(k)$) is defined. The operator determines the relevant score of the model.

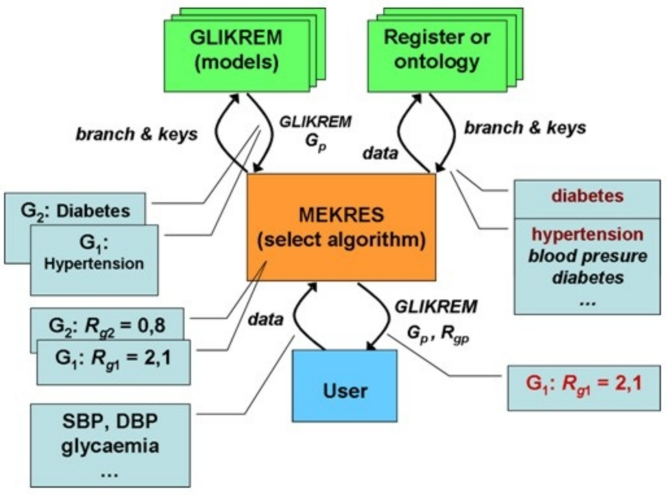

Figure 13. Medical knowledge representation system.

- e.g. the model G_1 has operator R_{g1} = 2,1 and the model G_2 has R_{g2} = 0,8.

- The model with the highest relevance value or a list of models ordered by the value of relevance is offered to participant.
    - e.g. the model G_1 (the highest relevance value) is offered to a patient.

Conclusion

The designed knowledge representation model (GLIKREM) based on the GLIF3.5 specification is the universal tool for the formalization of knowledge stored in a free text form (e.g. medical guidelines). The XML representation of the graphic model and the creation of the data interface in a paramodel form makes it possible to use it in different types of applications and to connect to real data from different sources (e.g. XML or CSV files). The definition in XSLT file form is used for the transformation of input data and the graphic model too.

The main extensions and benefits of the designed model are as follows:

- Our own formal representation of the graphic model in XML. The result of this representation is a XSD (XML Schema Definition).
- A more accurate definition of decision criteria and their evaluation in a three value logic.
- A definition of priority attribute for a reduction of a number of needed option evaluations in decision steps.
- A design of a paramodel (in XML) which is used as an interface between GLIKREM and real patient data in a clinical information system.
- An extension of GLIKREM by key attributes. This extension is used in a Medical Knowledge Representation System.

Acknowledgment

The paper is supported by the grant project No. AV0Z10300504 and No. 1M06014 of Ministry of Education, Youth and Sports CR.

References

[1] Peleg M., Ogunyemi O., Tu S., et al.: Using features of Arden Syntax with object-oriented medical data models for guideline modeling. Proc AMIA Symp. 2001:523-7.

[2] Peleg M., Tu S.W., et al.: Comparing Computer-interpretable Guideline Models: A Case-study Approach. The Jurnal of the American Medical Informatics Association, 2003 Jan-Feb; 10(1): 52-68.

[3] Shahar Y., Miksch S., Johnson P.: The Asgaard Project: a task-specific Framework for the application and critiquing of time-oriented clinical guidelines. Artif Intell Med 1998;14:29-51.

[4] Tu S.W., Musen M.A.: A flexible approach to guideline modeling. Proc AMIA Symp 1999:420-424.

[5] Tu S.W., Musen M.A.: From guideline Modeling to guideline execution: Defining guideline-based decision-support Services. Proc AMIA Annu Symp 2000:863-867.

[6] Quaglini S., Stefanelli M., Lanzola G., Caporusso V., Panzarasa S.: Flexible guideline-based patient careflow systems. Artif Intell Med 2001;22:65-80.

[7] Johnson P.D., Tu S.W., Booth N., Sugden B., Purves I.N.: Using scenarios in chronic disease management guidelines for primary care. Proc AMIA Annu Fall Symp 2000:389-393.

[8] Gennari J.H., Musen M.A., Fergerson R.W., Grosso W.E., Crubezy M., Eriksson H., et al.: The Evolution of Protégé: An Environment for Knowledge-Based Systems Development. Int J Hum Comput Stud. 2003;58(1):89-123.

[9] Bury J., Fox J., Sutton D.: The PROforma guideline specification language: progress and prospects. Proceedings of the First European Workshop, Computer-based Support for Clinical Guidelines and Protocols (EWGLP 2000), 2000.

[10] Ohno-Machado L., Gennari J.H., Murphy S.N., Jain N.L., Tu S.W., Oliver D., et al.: The GuideLine Interchange Format: A model for representing guidelines, Journal of the American Medical Informatics Association. 1998. ISSN 1067-5027.

[11] Zeng Q.: Guideline Interchange Format 3.5 technical specification. Available at: http://www.glif.org - Peleg, Boxwala, et al.

[12] Boxwala A.A., Peleg M., Tu S.W., Ogunyemi O., Zeng Q., Wang D., et al.: GLIF3: A Representation Format for Sharable Computer-Interpretable Clinical Practice Guidelines. J Biomed Inform. 2004;37(3):147-61.

[13] Sordo M., Boxwala A., Ogunyemi O., Greenes R.: Description and Status Update on GELLO: a Proposed Standardized Object-oriented Expression Language for Clinical Decision Support. Medinfo; 2004; 2004. p. 164-8.

[14] Laleci G.B., Dogac A., et al.: SAPHIRE: A Multi-Agent System for Remote Healthcare Monitoring through Computerized Clinical Guidelines. online at http://www .srdc.metu.edu.tr/webpage/projects/saphire/

[15] Buchtela D., Peleska J., Vesely A., Zvarova J.: Medical Knowledge Representation System, 21st International Congress MIE 2008 in Göteborg, 2008, Sweden.

[16] Buchtela D., Peleska J., Vesely A., Zvarova J.: Method of GLIF model Construction and Implementation, The XIX International Congress of the European Federation for Medical Informatics. 2005.

[17] Peleška J., Anger Z., Buchtela D., Šebesta K., Tomečková M., Veselý A., Zvára K., Zvárová J.: Formalization of Medical Guidelines. European Journal for Biomedical Informatics. 2005. Prague.

[18] Veselý A., Zvarova J., Peleška J., Buchtela D., Anger Z.: Medical Guidelines Presentation and Comparing with Electronic Health Record. International Journal of Medical Informatics 75 (3-4). 2006. 240-245

Contact
Ing. David Buchtela
Centre of Biomedical Informatics
Institute of Computer Science AS CR
Pod Vodárenskou věží 2
Prague 8 182 07
Czech Republic
e-mail: buchtela@euromise.cz