# Semantic decision trees editing for decision support with KcatoS - Application in oncology

Thomas Meilender, Jean Lieber, Fabien Palomares, Nicolas Jay

# Semantic decision trees editing for decision support with KCATOS – Application in oncology

Thomas Meilender[1,2], Jean Lieber[2], Fabien Palomares[1], and Nicolas Jay[2]

[1] A2ZI - 61 ter rue de Saint-Mihiel - 55200 Commercy
{thomas.meilender,fabien.palomares}@a2zi.fr
[2] UHP-Nancy 1 – LORIA (UMR 7503 CNRS-INPL-INRIA-Nancy 2-UHP)
{thomas.meilender,jean.lieber,nicolas.jay}@loria.fr

**Abstract.** During the last two decades, the interest for computer-interpretable guidelines has kept growing to become a major issue in medical informatics. Clinical guidelines usually contain decisional knowledge that can be represented by decision trees. This paper presents KCATOS, a semantic decision tree editor, which provides a collaborative tool to simplify knowledge acquisition. Using a simple graphical language, KCATOS allows exporting decision trees to formalised knowledge, by proposing an original export algorithm to OWL. Easy to integrate in web applications, KCATOS is part of a larger work about collaborative editing of clinical guidelines. The overall objective is to provide tools to assist editing and storage of decisional knowledge in the field of oncology.

## 1 Introduction

Knowledge acquisition is a well-known bottleneck in knowledge management. Experts have to provide accurate models of a domain by using complex formalisms. It appears that knowledge editing would be simplified if there existed a simple formalism that both experts and machines would understand.

As in most of medicine areas, oncology experts rely upon a particular kind of knowledge called decision knowledge. Decision knowledge associates decisions (or, at least, recommendations) to situations. Also known as medical guidelines, medical decision knowledge is often formalised as decision trees that can be understood by a machine if they are well-formed.

Oncolor is an association gathering physicians from the French region of Lorraine that are involved in oncology[3]. One of its roles is the creation of clinical practice guidelines (CPGs) that are defined in [7] as "systematically developed statements to assist practitioner and patient decisions about appropriate health care for specific clinical circumstances." On its static website, Oncolor provides to practitioners more than one hundred CPGs. This base is written in HTML and contains a lot of decision trees drawn with dedicated graphics standards. Because of the large amount of data and the importance of keeping them up to

[3] http://www.oncolor.fr

date, maintenance is a hard task. Moreover, the lack of semantic information associated to them makes this knowledge unavailable for an automatic use.

This paper presents KCATOS, a tool that aims at facilitating maintenance and bringing a semantic layer to the data. KCATOS proposes a semantic graphics editor allowing to draw decision trees in a simple language and export them into machine-understandable pieces of knowledge thanks to a translation algorithm. The resulting formalisation can be exported in OWL.

After the presentation of the context of application in Section 2, Section 3 describes the decision tree language of KCATOS and the translation from this language to OWL. The editor is presented in Section 4 with its user interface and its embedding in semantic web, particularly with the knowledge server KOWL and the semantic form editor EDHIBOU. An evaluation based on existing data transformation and Oncolor feedback is proposed on Section 5. Section 6 draws a conclusion and several ongoing and future work.

## 2 Context of application

### 2.1 Oncolor guidelines and The Kasimir project

Oncolor has edited 144 guidelines giving recommendations for the cares of many different cancer localisations (for example "cervical cancer limited to the cervix") or a few more general types of care related to oncology such as "dental cares". Typically, guidelines are structured documents that are composed of various kinds of contents: texts, decision trees, medical classifications, pictures, references, and glossaries. They are published on the Oncolor website and are available for both practitioners and patients. As medical knowledge is critical and continually evolving, guidelines have a limited lifetime and must be kept up to date. Oncolor constrains their updates to every two years.

A guideline update can be viewed as a workflow, where a guideline is a set of data that has to be modified by several users at different times. From an organisational point of view, Oncolor names a coordinator who has to validate each step of the process. This latter invites a committee of domain experts to check and update guidelines when needed. Then, the resulting document is validated by a larger regional audience seminar and can be possibly sent back to the committee. After this agreement, Oncolor members propose a new edited version that will be online if accepted by the coordinator.

At this point, numerous needs have emerged for Oncolor. One of them is the need for a collaborative tool to allow the committee to exchange during the first step of the update. At present, corrections of committee experts are provided to the coordinator as a list of written notes that have to be adapted to fit to the guideline standard. It would be easier if experts could directly change the guideline. That means finding an editing tool that would be simple enough to be used by specialists with limited skills in computer science. Moreover, keeping a historic of guideline modifications would be helpful. Another need comes from the final guideline editing. Oncolor does not have any webmaster in its staff so maintaining HTML document is a hard task.

Oncolor is involved in a research project called Kasimir, which aims at providing tools for decision knowledge management and decision support in oncology by exploiting CPGs. Started in 1997, Kasimir is a multidisciplinary project involving industrial (A2ZI) and academic (LORIA, CNAM Laboratory of Ergonomic) partners. Kasimir led to the development of software such as the knowledge server KOWL, and the instance editor EdHibou [3]. But, until now, most Oncolor guidelines are not formalised so the tools miss real knowledge to be used in a clinical context.

### 2.2 Towards a semantic wiki

Conceived in 1995 by Ward Cunningham, wikis are websites for creating and collaborative editing of content in a simple way [6]. Traditional wikis are usually based on a set of editable pages, organized into categories and connected by hyperlinks. Semantic wikis are born from the merging of wikis and the semantic web. Berners-Lee and Fischetti define them as improved wikis by the use of semantic web technologies [2].

Semantic wikis can be a solution to both Oncolor and Kasimir needs. They provide a collaborative editing tool with numerous features such as software versioning or layout editor. Moreover, with the semantic layer, Kasimir will have access to new knowledge formalisation techniques. Semantic wikis also bring new perspectives for indexing and mining CPGs.

The only elements of CPGs that cannot be directly edited in a wiki are decision trees. No semantic wikis provide that kind of extension. That is why KcatoS has been created. It also allows to formalise trees and to deal with decision knowledge in semantic web applications.

## 3 KcatoS framework

### 3.1 KcatoS decision tree language

**General presentation.** KcatoS decision tree language is a graphical representation based on a small set of geometrical shapes connected by directed edges. In this way, shapes are considered as nodes of a simple decision tree. From a semantic point of view, there are several kinds of node, each one with its own meaning as shown in Figure 1.

This representation was directly inspired from graphics standards of our partner Oncolor. Indeed, Oncolor CPGs use visual representations that can be considered as trees for most of them. An advantage to use these graphics standards is that Oncolor experts already know them. We want to preserve Oncolor's graphics semantics in order to facilitate future use of CPGs by physicians.

**Syntax.** To avoid ambiguities and insure consistency, classical syntactical rules of trees are used:

- a tree has exactly one root;

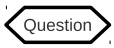
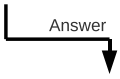
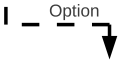
| Shape | Comments |
|---|---|
|   | Rounded rectangles represent *medical situations*. A medical situation can be defined by a patient state described by a set of variables such as medical exam results, physiology, etc. |
|   | *Recommendations* are highlighted by regular rectangles. They contain the advice that the CPGs would give the practitioner. The rectangle color has also a meaning: it specifies the kind of recommendation, for example surgery, specific treatment, or chemotherapy. |
|   | Hexagons represent *questions* that will help to describe the situation in order to define the right recommendation. |
|   | Ellipses are *links* from/to another tree: this enables to draw a big tree in several web pages, each of them containing smaller trees. |
|   | Edges are shown by simple arrows. When they connect a question to another node, links are typed, which means that they contain an *answer* to the previous question. |
|   | *Options* are particular kinds of edge: they correspond to therapeutic options which can be applied to a situation. From a medical point of view, it means that a choice does not depend on particular parameters, but let practitioners choose directly the preferred option. |

**Fig. 1.** Shapes and their meanings.

- nodes are connected by directed edges;
- texts on edge represent transition conditions, and may contain simple formulas: `AND`, `OR`, `NOT` are the only recognised boolean operators.

Moreover, a few rules are added to guarantee a correct semantics for the trees:

- the root is necessarily a situation or a link from another tree;
- a question has at least one answer, and every edge that follows a question must contain an answer;
- a text on an edge is an answer if the edge follows a question;
- a node may have several parents but directed cycles are forbidden, *i.e.* a node cannot appears more than once in a path;
- a node can be connected to its sons only by using edges or options;
- as a situation can have more than one recommendation, some recommendations can be gathered.

An example of syntactically correct tree is shown in Figure 2.

**Semantics.** A KCATOS decision tree can be exported to OWL, as shown in the next section. Its semantics is the semantics of the OWL knowledge base.

### 3.2 Export from KCATOS to OWL

In this paper, the description logics $\mathcal{SROIQ}(D)$, which is equivalent to OWL DL 2, is used to describe the axioms.

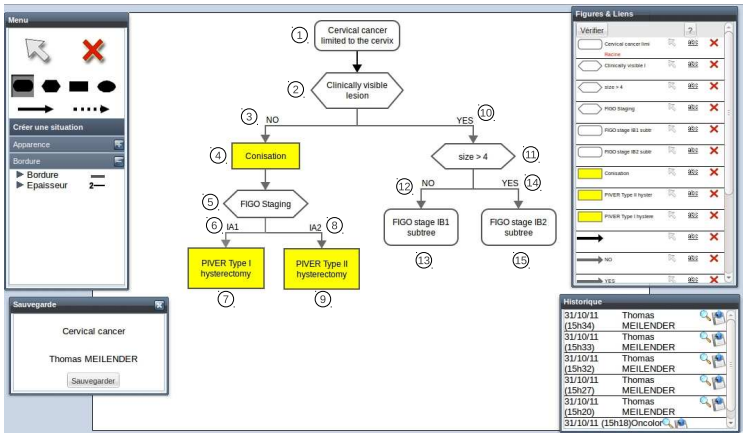



**Fig. 2.** An excerpt of the Oncolor treatment guideline for "cervical cancer limited to the cervix" edited with KCATOS editor.

KCATOS 's export algorithm defines two classes: Situation and Recommendation. The first one represents the situation (e.g., in the oncology application, some patient description) while the second one represents the description of the decision proposed by the system. These classes are linked in this way:

$$\texttt{Situation} \sqsubseteq \exists \texttt{hasRecommendation}.\texttt{Recommendation}$$

This means that for each situation $\sigma$ ($\sigma \in \texttt{Situation}^{\mathcal{I}}$) there is a recommendation $\rho$ ($\rho \in \texttt{Recommendation}^{\mathcal{I}}$) that is associated to $\sigma$ ($(\rho, \sigma) \in \texttt{hasRecommendation}^{\mathcal{I}}$). The property `hasRecommendation` relates a situation to a recommendation. It uses the class `Situation` as domain and the class `Recommendation` as range.

New subclasses for `Situation` and `Recommendation` are defined by the translation process. For example, let us consider a patient who has a headache and whose recommendation is to have some aspirin. Classes `PatientWithHeadache`, subclass of `Situation`, and `AspirinPrescription`, subclass of `Recommendation`, can be defined in this way:

$$\texttt{PatientWithHeadache} \equiv \texttt{Situation} \sqcap \exists \texttt{hasSymptom}.\texttt{HeadacheSymptom}$$

$$\texttt{AspirinPrescription} \sqsubseteq \texttt{Recommendation}$$

Then, the following formula formalises the (controversial) sentence "Each patient with a headache has to be prescribed aspirin.":

$$\texttt{PatientWithHeadache} \sqsubseteq \exists \texttt{hasRecommendation}.\texttt{AspirinPrescription}$$

### 3.3 Translation rules

A tree is read using depth-first search. Each node is transformed using the rules defined below which take into account its ancestors.

A whole example of export from graphical decision tree to OWL is shown in Figure 3.

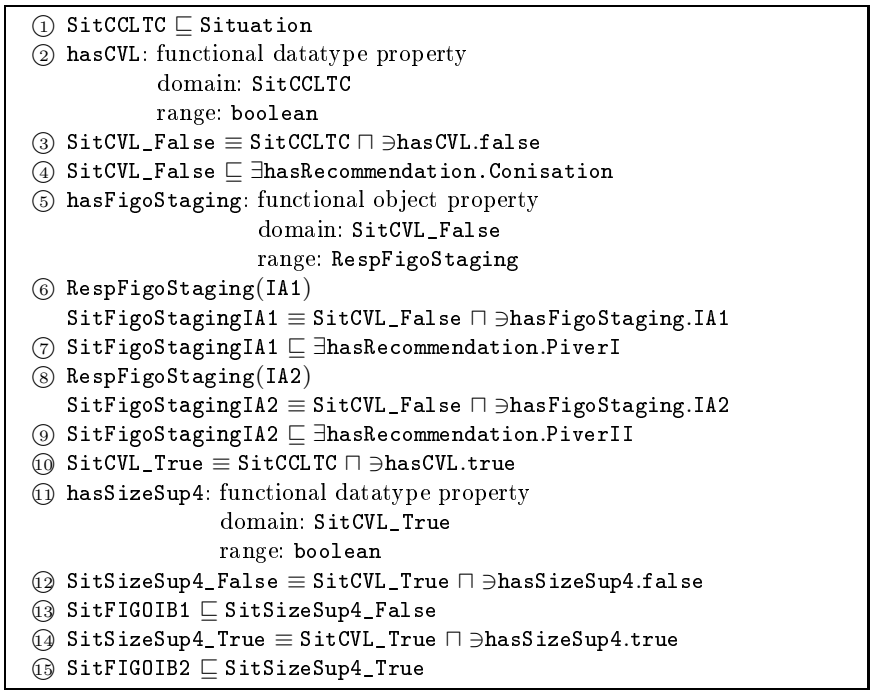



**Fig. 3.** OWL translation of decision tree edited in Figure 2.

**Situations.** A situation shape allows to create a class `Sit_Y` that is a subclass of `Sit_X`, the nearest subclass of `Situation` (*i.e.* the the subclass of the parent node or parent edge, if any).

$$\texttt{Sit\_Y} \sqsubseteq \texttt{Sit\_X}$$

**Recommendations.** A recommendation shape shows that a situation class `Sit_X` is linked to the recommendation `Reco1` by the previously defined property `hasRecommendation`.

$$\texttt{Sit\_X} \sqsubseteq \exists \texttt{hasRecommendation.Reco1}$$

**Questions.** A question shape introduces a new functional property `hasAnswer` having `Sit_X`, the nearest subclass of situation, as domain. If the answers are "yes" and "no", `hasAnswer` is a datatype property with a boolean range. Else, it

is an object property having a new class `AnswerQuestion` as range.

$$\begin{aligned} \texttt{hasAnswer} : & \textit{functional property} \\ & \textit{domain} : \texttt{Sit\_X} \\ & \textit{range} : \texttt{boolean}\ \textit{or}\ \texttt{AnswerQuestion} \end{aligned}$$

**Links.** Ellipses permit to do links between trees and appear as root or as leaf. It means that the situation `Sit_X` described in the first trees is equivalent to the initial situation `Sit_Y` described in the second tree.

$$\texttt{Sit\_Y} \equiv \texttt{Sit\_X}$$

**Edges.** An edge contains an answer `ANSWER` to the question `hasAnswer` it directly follows. It introduces a new subclass of `Sit_X`, by specifying the property value.

$$\texttt{Sit\_Y} \equiv \texttt{Sit\_X} \sqcap\ \ni \texttt{hasAnswer.ANSWER}$$

**Options.** This arrow can be viewed as a particular kind of edge. But, by contrast to classical edges, they define a situation by filling the property `hasOption` and create `TherapeuticOption` instances.

$$\texttt{Sit\_Y} \equiv \texttt{Sit\_X} \sqcap\ \ni \texttt{hasOption.OPTION}$$

## 4 Embedding KCATOS in the semantic web

### 4.1 Technologies in use

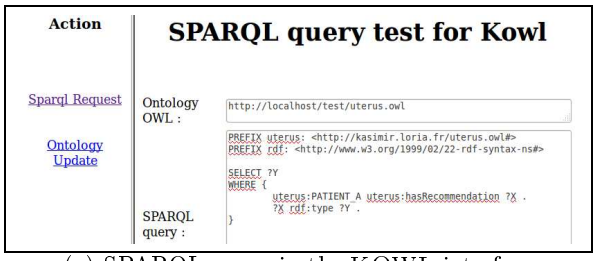
KCATOS decision tree editor is a web-based application using Google Web Toolkit[4] (GWT) that allows to create complex Ajax applications. A few additional APIs dedicated to GWT are used to manage the interface. Drawing capabilities rely on SVG and JavaScript technologies while OWL export is done thanks to OWL API [5]. Thus, KCATOS is open to collaborative work and web services. Its framework can be integrated in most of content management system: some tests have already been successfully done with MediaWiki[5].

A syntactic module can be used to check if the edited tree respects rules defined in Section 3. Included in the interface, the module allows to validate trees step by step while drawing, by identifying shapes with mistakes. As an output, different formats are proposed: bitmap (PNG and JPG), Vector graphics (SVG), and ontologies (OWL). Moreover, KCATOS includes its own version systems. As each tree is kept on a distant server, modifications are saved. For the present, only few functions dealing with history are available: previous versions of a tree can be viewed and restored with some informations about author and date. However, some information is saved into XML files that will allow to add functionalities such comparison of versions and merging algorithms. Those improvements are planned to be part of our future work. A screenshot of a stand-alone version is shown in Figure 2. It will be shortly available under a free licence (LGPL).

[4] `http://code.google.com/intl/en/webtoolkit/`
[5] `http://www.mediawiki.org`

### 4.2 Querying knowledge with semantic web applications

Export to OWL allows the KCATOS use in the semantic web. Previously created in the Kasimir project, KOWL is a knowledge web server. KOWL can read and edit a remote OWL file during a session. It supports SPARQL [9] queries through a simple HTML interface. Editing relies on Jena API[6] while inferences are done with the Pellet reasoner [11].




(a) SPARQL query in the KOWL interface.

| Y |
|---|
| http://kasimir.loria.fr/uterus.owl#Conisation |
| http://www.w3.org/2002/07/owl#Thing |
| http://kasimir.loria.fr/uterus.owl#Recommendation |

(b) XML answer to the previous query.

**Fig. 4.** An example of SPARQL query (a) to get recommendation classes (b).

A KOWL query to the knowledge base created in Figure 3 is shown in Figure 4. The goal of this example is to get a recommendation for a patient `PATIENT_A` who has a cervical cancer limited to the cervix without any visible lesion. First, two instances are created, an instance `PATIENT_A` in class `Situation` and an instance `MY_RECOMMENDATION` in class `Recommendation` which are linked by the property `hasRecommendation`. Then, the property `hasCVL` is set to `false` for the instance `PATIENT_A`. Finally, a SPARQL request is sent to get classes of `MY_RECOMMENDATION` (Figure 4(a)). Answer can be seen in Figure 4(b). Three classes are given: classes `Thing` and `Recommendation` can be easily deduced and class `Conisation` is inferred. It means that a conization (i.e., a special kind of biopsy of the cervix) is recommended for `PATIENT_A`, which corresponds to the recommendation given in the previous decision tree.

Interaction capabilities with EDHIBOU [1] are another example of KCATOS use in the semantic web context. EDHIBOU is a framework that aims at providing a user interface for the Kasimir project. It allows to describe a medical situation step by step by using questions described as hexagons in decision trees. Questions appear in a HTML form, following the order of the tree. According to user answer, recommendations can then be deduced and shown.

[6] `http://www.openjena.org`

Technically, EdHibou edits OWL ontologies. Its principle is to create an instance and to consider related properties as questions. Editing and inference in EdHibou rely on the previously presented knowledge server KOWL. From a visual point of view, EdHibou offers many customisation possibilities. Multiple configurable views are proposed in order to keep a check on element evolutions depending on user choices. In this way, a view that shows the inferred classes of the created instance can be made. This view is kept up to date at every user interaction. Graphical widget contained in forms can also be customised for a particular application. All those graphical component customisations can be done by editing an interface-dedicated ontology.

Combined to KcatoS, EdHibou uses exported OWL files. It creates an instance of `Situation` and an instance of `Recommendation` and links them with the property `hasRecommendation`. A view is launched to visualize the label of the classes of the instance of `Recommendation`. EdHibou transforms into HTML form widgets each property which has as domain the classes of the instance of `Situation`. While filling the form, values are given to properties in the ontology for created instance of `Situation`. As this last instance becomes more specific, more specific classes of the instance of `Recommendation` are inferred and their labels are shown in the recommendation part of the view. An example of such use is shown in Figure 5.

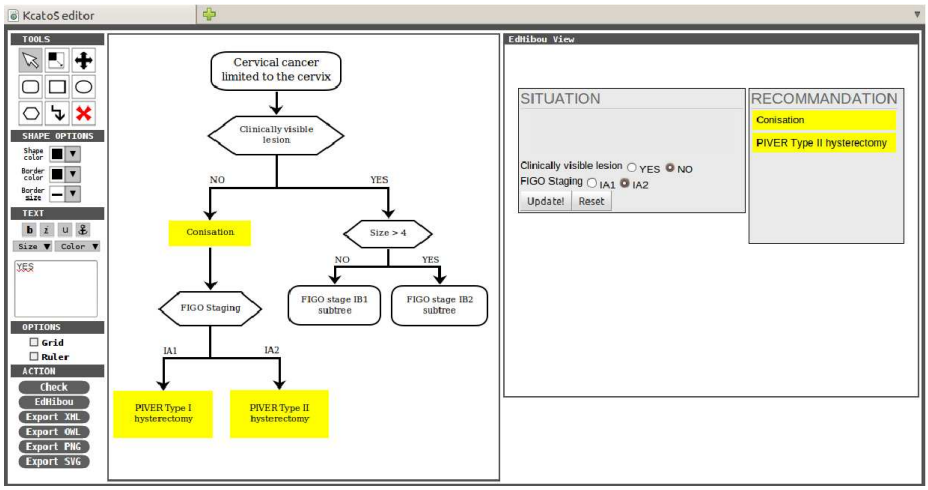



**Fig. 5.** KcatoS combined with EdHibou.

### 4.3 Using KCATOS in semantic wiki context

KCATOS has been integrated in a semantic wiki, OncologiK[7] [8]. A semantic wiki is similar to a traditional one in the sense that it is a website where contents are added by users. This content is organized into editable and searchable pages, accessible to all users. However, unlike traditional wikis, semantic wikis are not limited to natural language text. They characterize resources and links between them. This information is formalized and thus become usable by a machine, through processes of artificial reasoning.

From a technical point of view, OncologiK uses as a wiki engine Mediawiki and its extension Semantic Mediawiki (SMW) [12]. They allow the use of semantic applications such as semantic forms and SPARQL access. KCATOS has been integrated as an extension and uses templates and parsers functions of SMW to import knowledge about trees. In this way, trees are indexed in the semantic wiki. For example, the system allows to find by a query which trees are concerned about digestive tract.

## 5 Evaluation

### 5.1 Dealing with existing trees

Oncolor experts have already written 144 guidelines that contain more than 600 decision trees. Most of them have a size equivalent to the tree shown in Figure 2. Unfortunately, throughout successive updates, medical experts did not always take care of respecting neither classical rules of decision trees, nor graphical standards. Resulting trees are still readable and understandable by specialists but are far from being systematically recognizable. KCATOS syntax verification algorithms will not accept syntactically incorrect decision trees. After analysing 150 existing decision trees, it has appeared that only 44 could be considered as well-formed. By studying the causes of errors, frequent mistakes can be identified and easily corrected such as the absence of initial situation.

However, 62 trees presented mistakes that need specific corrections. According to the critical nature of data, these trees have to be corrected by oncology experts, and may sometimes have to be rewritten in a correct manner. Considering the nature of errors, some have been corrected after this evaluation by adding the "Option" shape presented in Section 3.1 that did not exist previously. Other minor corrections were made such as adapting a shape or a color, drawing or completing missing transitions, etc. An example of complex correction involved the presence of a directed cycle. It was due to the meaning of the described process: a medical exam has to be done several times until the result is satisfying. Until now, no OWL formalisation has been found to express that kind of knowledge.

[7] `http://www.oncowiki.fr`

### 5.2 Analysing existing trees expressiveness

If KcatoS allows to formalise most of the knowledge contained in the trees, an extended expressiveness is needed in a few cases. The previous example of forbidding directed cycle revealed a limit of the KcatoS language. Moreover, decision knowledge may also depend on various factors such as time, computation of a score, or a complex set of criteria. Including those particular kinds of transition would mean extending KcatoS vocabulary. The risk would be to make the system more complex and increase the barriers to medical experts. From a formal stand point, they refer to complex subjects already tackled in the literature. Dealing with time in OWL is made possible by OWL-Time [4]. Concerning sets of criteria, they are present when the decision depends on many factors, clinical common sense or practitioner experience. In some cases, it seems that a fuzzy logic approach would be helpful.

### 5.3 Feedbacks from medical experts

KcatoS has already been presented to Oncolor experts and has been in use since the end of August. Medical experts view of decision is a bit different from knowledge engineer ones. They consider decision trees as a compromise between a logical view and a graphical representation that is clear for clinicians. From this point of view, ambiguities are minors problems that most of the time can be solved using domain knowledge. Clinicians have this knowledge, but it is hard for automatic systems to get them. That is why new work is planned in collaboration with Oncolor experts to make explicit some parts of trees. The difficulty will be to extend trees while keeping a simple representation that can be understood quickly by physicians.

## 6 Discussion and Future work

Clinical guidelines generally contain decision knowledge that can be represented by decision trees. In this paper, the framework KcatoS has been presented. Based on OWL and using a simple language, KcatoS provides a collaborative editor, easy to integrate in semantic web applications.

KcatoS has been presented to Oncolor staff. At this point, exchanges with experts led us to enlarge the vocabulary by adding two shapes: links and option. Moreover, recommendation shapes have specific colors which precise their meaning. This particularity will be added in OWL export by defining subclasses of `Recommendation`. Transitions can also be improved by taking into account temporality and scoring. Another problem is that, in some cases, some directed cycles represented in guidelines cannot be a part of well-formed trees. In order to solve this issue, KcatoS has to take more general structures into account.

KcatoS is a part of a larger work about collaborative editing of clinical guidelines. The overall objective is to provide tools that can assist domain specialists to edit and store decisional knowledge. In order to reuse that knowledge

in knowledge-based systems, the semantic wiki technology is used. It will bring us tools to facilitate alignment with existing biomedical ontologies and thesaurus such as SNOMED [10]. Annotating guidelines with those ontologies will bring several benefits. At first, it will allow to deal more easily with other semantic web applications and exchange data and services. Then, practitioners using SNOMED could have a significant help while encoding medical records.

***Acknowledgements.*** The authors wish to thank the reviewers for their helpful comments.

## References

1. F. Badra, M. D'Aquin, J. Lieber, and T. Meilender. Edhibou: a customizable interface for decision support in a semantic portal. In C. Bizer and A. Joshi, editors, *International Semantic Web Conference (Posters & Demos)*, volume 401 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.
2. T. Berners-Lee and M. Fischetti. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. Harper San Francisco, 1st edition, 1999.
3. M. D'Aquin, C. Bouthier, S. Brachais, J. Lieber, and A. Napoli. Knowledge Editing and Maintenance Tools for a Semantic Portal in Oncology. *Int. Journal on Human-Computer Studies*, 62:619–638, 2005.
4. J.R. Hobbs and F. Pan. Time Ontology in OWL. Working draft, September 2006.
5. M. Horridge and S. Bechhofer. The OWL API: A Java API for OWL ontologies. *Semantic Web*, 2(1):11–21, 2011.
6. B. Leuf and W. Cunningham. *The Wiki Way: Collaboration and Sharing on the Internet*. Addison-Wesley Professional, April 2001.
7. K.N. Lohr, M.J. Field, United States., and Institute of Medicine (U.S.). *Clinical practice guidelines : directions for a new program / Committee to Advise the Public Health Service on Clinical Practice Guidelines, Institute of Medicine ; M.J. Field and K.N. Lohr, editors*. National Academy Press, Washington, D.C., 1990.
8. T. Meilender, J. Lieber, F. Palomares, and N. Jay. From web 1.0 to social semantic web: Lessons learnt from a migration to a medical semantic wiki. In *ESWC 2012*. To be published, 2012.
9. E. Prud'hommeaux and A. Seaborne. Sparql query language for rdf (working draft). Technical report, W3C, March 2007.
10. P. Ruch, J. Gobeill, C. Lovis, and A. Geissbühler. Automatic medical encoding with snomed categories. *BMC Medical Informatics and Decision Making 2008*, 8(Suppl 1):S6, 2008.
11. E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, June 2007.
12. M. Völkel, M. Krötzsch, D. Vrandecic, H. Haller, and R. Studer. Semantic wikipedia. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 585–594, New York, NY, USA, 2006. ACM.