HAL
open science

Virtual Staff - Software Tool for Cooperative Work in a Health Care Network

Marek Ruzicka, Rose Dieng-Kuntz, David Minier

To cite this version:

Marek Ruzicka, Rose Dieng-Kuntz, David Minier. Virtual Staff - Software Tool for Cooperative Work in a Health Care Network. RR-5390, INRIA. 2004, pp.40. inria-00070613

HAL Id: inria-00070613

https://inria.hal.science/inria-00070613

Submitted on 19 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Virtual Staff - Software Tool for Cooperative Work in a Health Care Network

Marek Ruzicka, Rose Dieng-Kuntz, David Minier

N° 5390

Novembre 2004

THÈME SYMB

Rapport de recherche

ISSN 0249-6399

Virtual Staff - Software Tool for Cooperative Work in a Healthcare Network

Marek Růžička [1], Rose Dieng-Kuntz[2], David Minier

Thème SYMB – Systèmes symboliques
Projet ACACIA

Rapport de recherche n° 5390 – Novembre 2004 - 38 pages

Abstract: In the context of knowledge management for a healthcare network, we have developed a software tool called Virtual Staff enabling a cooperative diagnosis by network actors, visualization of their collective reasoning and preparation of repository of patient records which will be reused for decision support for new patient cases. This virtual staff relies on a medical ontology and enables users to cooperative building of graphs based on the SOAP (Subjective, Objective, Assessment, Plan) model used in medical community and on the QOC (Questions, Option, Criteria) model used by CSCW community for support to design rationale and decision-making support.

Keywords: ontologies, knowledge management, cooperative work, decision-making support, QOC, SOAP, healthcare network

[1] Marek.Ruzicka@sophia.inria.fr
[2] Rose.Dieng@sophia.inria.fr

Unité de recherche INRIA Sophia-Antipolis
2004, route des Lucioles, 06902 Sophia-Antipolis (France)
Téléphone : +33 (0)4 92 38 77 77 – Télécopie : +33 (0)4 92 38 77 65

Virtual Staff - Software Tool for Cooperative Work in a Health Care Network

Résumé: Dans le contexte de la gestion des connaissances pour un réseau de soins, nous avons développé un outil de logiciel appelé Staff Virtuel permettant un diagnostic coopératif par des acteurs du réseau, la visualisation de leur raisonnement collectif et la préparation d'une base de dossiers patients qui seront réutilisés pour l'aide à la décision sur de nouveaux cas de patients. Ce Staff Virtuel repose sur une ontologie médicale et permet aux utilisateurs la construction coopérative de graphes basés sur le modèle SOAP (Subjective, Objective, Evaluation, Plan) utilisé dans la communauté médicale et sur le modèle QOC (Question, Options, Critères) utilisé par la communauté « collecticiels » pour l'aide à la logique de conception et pour l'aide à la décision.

Mots clés: ontologies, gestion des connaissances, travail coopératif, aide à la décision, QOC, SOAP, réseau de soins.

1 Virtual Staff

1.1 Objectives of Virtual Staff

The *"Ligne de Vie"* (*Life Line*) project aims at developing a knowledge management tool for a health care network. Specialized in a particular domain or in a specific pathology, a health care network is a health network gathering all the actors intervening in the care or follow-up processes. The objective of the network is to ease (a) communication and collaboration among these actors in spite of their physical distance, (b) the regular follow-up of the patient et (c) the respect of best practices inside the network. The patient must be guided towards relevant medical actors, that must be informed about the patient's state and that may gather (sometimes virtually, through synchronous or asynchronous communication tools) in order to work on the patient's record. For example, cystic fibrosis demands daily cares throughout the patient's life and requires several kinds of professionals : paediatricians, physicians, gastroenterologists, chest specialists, physiotherapists, psychologists, dieticians, social workers, as well as the patient, his/her family, the school doctor or the job doctor. The network must ease knowledge sharing about the patient record, among all these actors from various competence domains, with a user-tailored presentation of the information.

For supporting such care network, we developed a software tool called *Virtual Staff,* which enables members to *visualize their collective reasoning*: in order to diagnose the pathology of the patient (according to the symptoms expressed by the patient, the observations or analyses of the doctor and the already known health problems of this patient), or in order to determine the best possible therapeutic procedures. This Virtual Staff should offer to the users *a service of support to cooperative reasoning*, during phases of elaboration of diagnosis or therapeutic decision and *a service of constitution of an organisational memory* (Dieng et al, 1999; Dieng-Kuntz and Matta, 2002) – the memory of decisions of the community constituted by the members of the care network.

In the hospital, the unity of location and of time allows the doctors to meet as a staff in order to discuss about the decisions to take. In a health care network, the Virtual Staff aims to be a collaborative work supporting tool, allowing the real time update and history of therapeutic decisions. As an *electronic board* where each one can note information readable by the other members of the team, it constitutes a discussion support that may be synchronous (if the participants take part to the discussion at the same time or in the same place) or asynchronous (if each one accesses it at the moment appropriate to him/her). Starting from the patient's health problems, the members of the team will formulate diagnostic hypotheses and proposals for a treatment. Via this Virtual Staff, the care team will connect the various elements of the patient record useful for the discussion, and thus will converge in an asynchronous way towards the definition of new health problems and of new therapeutic actions. The formulation of diagnostic hypotheses is a priori reserved to the medical actors, whereas the discussion on the treatment could sometimes imply non medical professionals (for example, a welfare officer could emit arguments against the choice of a heavy treatment incompatible with housing conditions of the patient).

We suppose that several patient cases will be created *asynchronously* by *multiple users*. This requires to visualize and store all information about the patient case, including multi-user and temporal aspects. Another objective of Virtual Staff is to collect solved cases in a special *repository* and to reuse them for retrieving past patient cases similar to a new patient case to solve. This kind of reasoning is similar to case-based reasoning.

1.2 Virtual Staff and its associated components

Virtual Staff (VS) is not a single stand-alone application. It is not either only a graphical editor and visualization tool of patient cases. Besides VS, there are three other important components involved in preparation and reasoning of patient cases. First of all, there is the **Nautilus Data-**

base of electronic patient records (EPRs) serving as source of initial medical data for new patient cases, **Extended Nautilus Ontology** ensuring consistent terminology and structure of patient records and finally **CORESE** search engine used for querying Nautilus ontology and stored patient cases. Whole context is described on fig. 1.

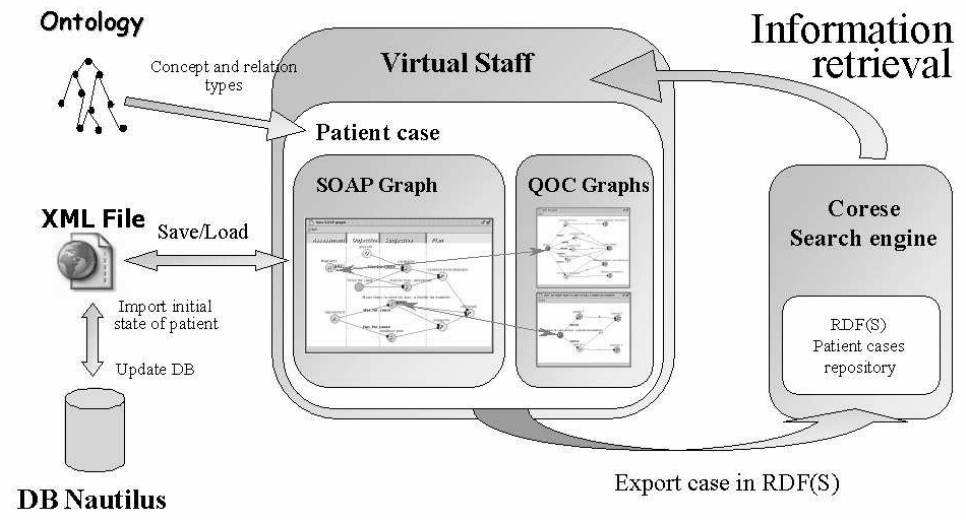


Figure 1. Overview of Virtual Staff and its associated components

1.2.1 Nautilus DB of electronic patient records

The ideas proposed by Nautilus SARL for launching the project Ligne de Vie stem from fifteen years of experiences in health domain and from contacts with a network dedicated to diabetes: Nautilus offers a software for management of electronic medical record, **Episodus**, relying on a problem-oriented vision of the patient records and articulated around the notion of "Life Line" enabling to represent the life of the patient from his/her birth till his/her death with all the health problems encountered by this patient. As long as the patient still suffers from a problem, this problem remains open. When the patient gets completely cured from it, the problem is closed. Figure 2. shows an example of such life line of a patient. Through Episodus, we could see evolution of all symptoms, diseases, treatments and other health care procedures (we will call them altogethar *concepts*). Major advantage of cooperation of Virtual Staff with Episodus should be in obtaining initial data about patient from his/her existing EPR in Nautilus database. It is useful to start a consultation with patient not with an "empty case", but with a case containing all mentioned concepts, which are still active at the time of the consultation.

When a patient consults his/her doctor for new symptoms, the physician will create an instance of Virtual Staff. The system will then initialize a patient case with all currently open pathologies and prescriptions for this patient, based on available information from data file imported through Episodus. Then the user can add new concepts in the case and reason about them in context of other existing concepts. If some new important concept is approved by this reasoning (or the user suggests that existing one should be changed), VS has to be able to store this information in data file readable by Episodus. Later, when the user of Episodus opens patients' case, (s)he will see also results of reasoning in Virtual Staff and s(he) can accept/reject these results.

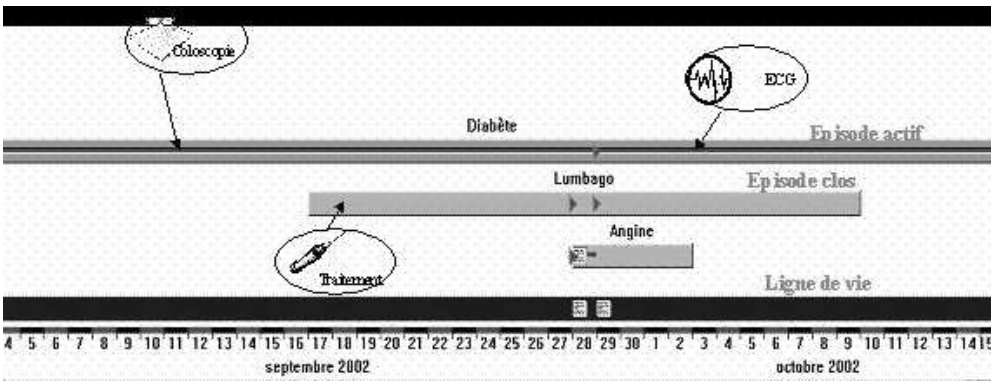


Figure 2. Episodus representation of life-line

1.2.2 Role of Nautilus Ontology and exchange format

Virtual Staff visualizes patient case as an oriented graph, where each node corresponds to one concept. Concepts should use not only standard medical names but they should be also taken from some hierarchy of medical terms. Therefore VS uses Extended Nautilus ontology as source of terminology. Linking of graph nodes to standard medical terms ensures easier understanding of the patient graph/case to other involved participants or even to newcomers. As Nautilus ontology is based on Nautilus database, concepts from this ontology are understandable by Episodus. This strongly simplifies cooperation between VS and Episodus, which would be almost impossible without using a similar ontology. Likewise nodes, arcs should by associated with relation types defined in the ontology. Unfortunately Nautilus Ontology did not offer the needed relation types, hence it was necessary to enrich it with a minimum set of relations[3]. This also points to flexibility of Virtual Staff. Everything is taken from the ontology and no concept/relation type is hard-coded, so it is possible to switch to another ontology represented in RDF(S) without modifying source code.

Even though both tools, Virtual Staff and Episodus, rely on terms taken from the Nautilus Ontology, to enable real communication between them, we had to develop a specialized exchange format. Through this format Episodus is sending initial information about patient to Virtual Staff and vice versa VS sends back update of data for Nautilus DB according to acquired results. This exchange format is written in XML and is fully compatible with Virtual Staff. Nautilus SARL is currently developing interface for Episodus for this exchange format. Detailed specification of this format is described in section 5.2.

[3] See Appendix B

Figure 3. Extract of Extended Nautilus Ontology

1.2.3 CORESE and Patient cases repository

CORESE (COnceptual REsource Search Engine) [4] (Corby et al, 2000; Corby and Faron-Zucker, 2002; Corby et al, 2004) is a semantic search engine dedicated to RDF(S) developed by the Acacia team. It is widely used within VS. Since the extended Nautilus ontology is represented in RDF(S), we can use Corese for finding requested concepts and relation types. While adding or updating nodes in already existing parts of the graph, the system can propose a list of possible concepts to help the user to find the correct one.

VS enables to store patient case not only in the previously mentioned XML exchange format, but also in RDF(S) file. Through saving cases in RDF(S) we can obtain repository of closed/solved patient cases which could be later queried by Corese search engine or analysed by statistics and data-mining methods. Querying repository by Corese could be extremely helpful for finding similar cases in order to compare new cases with solved ones. For example, a physician could query Corese about all the past patients for which a session of the virtual staff enabled to take a decision between a chemotherapy and an hormonotherapy, and to ask Corese to retrieve the (positive or negative) arguments that had been exchanged for reaching this decision..

[4] http://www-sop.inria.fr/acacia/soft/corese/

2 Models of SOAP and QOC graphs

2.1 Weed's SOAP Model

In Virtual Staff, the dependencies between the various diagnostic and therapeutic hypotheses can be represented through a graph using the concepts defined in the Nautilus ontology. The doctor will reason by linking the health problems to the symptoms, the clinical signs and the observations in order to propose health care procedures.
The Virtual Staff can thus rest on the SOAP model (Subjective, Objective, Assessment, Plan) dividing concepts into four major categories. In this model:

- the **S nodes** describe current symptoms and clinical signs of the patient,
- the **O nodes** describe analyses or observations of the physician,
- the **A nodes** correspond to the diseases or health problems of the patient,
- and the **P nodes** correspond to the procedures or action plans set up in order to solve the health problems.

This SOAP model is used in the medical community to structure a patient record. Therefore, its use to structure the doctor's reasoning - that relies on the same concepts - seems natural. Original Weed's SOAP model was used only for clustering notes into mentioned categories. In our approach we try to add dynamic form to this model and transcribe it into an oriented graph (see fig 4).

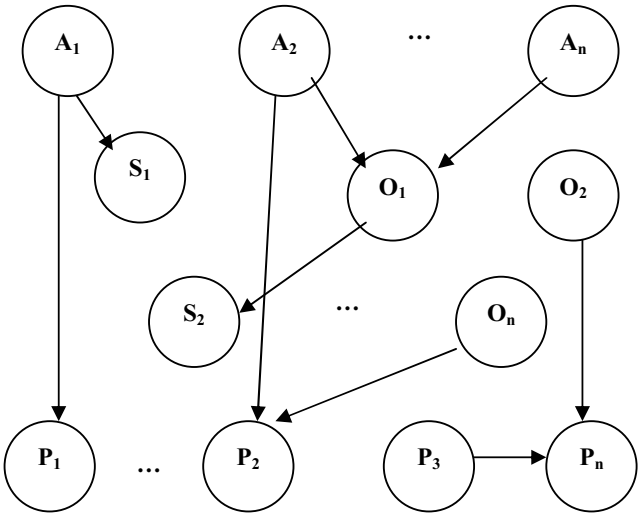


Figure 4. SOAP model as an oriented graph

When a new case is opened, the system will initialize a SOAP graph with some parts prepared according to information taken from XML data file: the initial A and P nodes automatically added in the graph are the pathologies and the care procedures already existing and open in the patient's life line. The doctor can then reason in order to add, in case of need, new A nodes (i.e. new pathologies diagnosed) or new P nodes (i.e. new action plans) in order to diagnose and treat the new health problem of the patient.
Each node on oriented SOAP graph (dynamical form of SOAP model) belongs to one of SOAP categories. At the same time each node is linked to a concept from Nautilus ontology. Root concept type of this ontology is divided into 9 major subtypes. It was possible to put no restriction

on combinations of SOAP categories and Nautilus concept types at the same node, but after closer examination of both classifications, we can find a meaningful mapping between them (see table 1), which is also used in VS.

Table 1.

SOAP Category	Nautilus concept type	Nautilus code
Subjective	Symptômes (Symptoms)	S
Objective	Examens de labo (Laboratory tests)	T
Assessment	Agents pathogènes (Pathogenous agents)	R
	Corps étranger (Foreign body)	C
	Malformation	E
	Pathologies	P
	Psy (Psychological problems)	Y
Plan	Traitement (Treatment)	N
	Geste diagnostique ou thérapeutique (Diagnostic or therapeutical gesture)	G

2.2 QOC model (Question-Options-Criteria)

Sometimes, the doctor may need to visualize all possible solutions and the arguments in their favour or against them. The QOC model (Question Options Criteria), used by CSCW community for support to decision-making or for design rationale in a project, can then be useful. In this model, a question Q corresponds to a problem to solve. To solve the question Q, several Options are thought out, with, for each option, the criteria in its favour and the criteria against it: each option is thus connected positively or negatively to criteria. The QOC graph is reduced to a tree if no criterion is linked to several options.
Two types of questions are possible for the Virtual Staff:

- *Diagnosis of a pathology* (i.e. find the right A in the SOAP model): Which pathology explains the clinical signs of the patient?
- *Search of a prescription* (i.e. find the right P in the SOAP model): Which action plan will enable to treat the diagnosed pathology?

In the Virtual Staff, among the criteria to be satisfied, there are the patient's symptoms and the doctor's observations: for a question about the patient's pathology, each possible option will be linked by a positive influence link to the symptoms and observations compatible with this option, and by a negative link to the symptoms or observations rather incompatible with this pathology. The criteria will thus consist of S or O nodes of the SOAP model but they may also sometimes correspond to A or P nodes, if some diseases are incompatible or if some care procedures are exclusive.

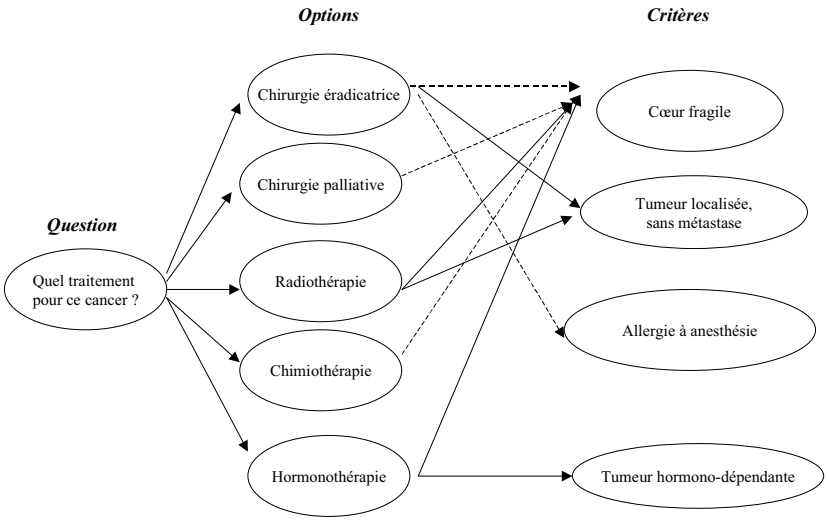


Figure 5. Example of QOC graph

For support to decision on a treatment to cure the diagnosed pathology, the options will be the various possible treatments, each one connected by a positive link to the criteria encouraging to choose it and by a negative link to the criteria inciting to reject it. For example, fig. 5 shows a QOC graph for choosing between an hormonotherapy, a chemotherapy, a radiotherapy, a palliative surgery and an eradicating surgery in order to treat a cancerous tumour, taking into account several criteria such as some characteristics of the tumour (e.g. hormone-dependent tumour, localised tumour without metastasis) or of the patient (e.g. fragile heart, allergy to anaesthesia).

2.3 Interaction between SOAP and QOC graphs

Since we want to use two different kinds of graphs in complementary way, we have to specify how these graphs will be connected. Complex patient case can be easily seen on the SOAP graph. On the other hand the QOC graph shows discussion about one specific health problem. We can simply imagine that for each SOAP node we could have one QOC graph discussing this node in more detail. Unfortunately this vision of connection between graphs is too general and it does not give answers to important questions like “when the user should start QOC”, “what happens when QOC is decided”, etc.

The essential part of QOC graph is its “Question”, which is not only “question part” of graph, but also reason for starting this graph. From general point of view, the user may open a QOC whenever (s)he is not certain of “something” inside the patient case and wants to start a discussion. There are three possible situations/questions described below.

1) There is something missing in SOAP graph:

The user knows (or thinks) that there should be new node in the SOAP graph, but (s)he is not sure about the correct concept for it. In this situation s(he) starts a QOC for new SOAP node (temporary literal node) and when the decision is made, the decided option from the QOC graph takes place instead of the literal node. Changing and removing decision later will also influence concept type of the SOAP node.
In Virtual Staff temporary literal node will be labelled with "?" symbol. After picking one of options in QOC graph, SOAP node is automatically transcribed by selected option (concept type and label are copied to the SOAP node). The user cannot change directly the SOAP node, s(he) must enter the QOC graph and alternatively pick another decision.

2) User knows the category, but not the details:

In this case the Question node is not a literal node, but it already has an associated concept from the Nautilus ontology. Options are subtypes of Question concept defined in the ontology. The user for example knows that the patient needs chemotherapy, but (s)he is not sure which type of chemotherapy. Using a QOC, (s)he can invite other participants to discuss proper subtype. But at the same time, in the complex SOAP graph, other users can already see SOAP node for chemotherapy and its relations to other concepts.

3) User is not sure about existing concept

This case is little bit different, since the decision about SOAP concept was already made. This situation appears when the user is not absolutely sure about his/her previous decision and (s)he wants to ask others whether they agree with it (it means that level of certainty on a node is not 100%). For this kind of QOC it does not make sense to *remove decision*. Users can only add new options and possibly switch to one of them.
Also implementation in VS is different than in previous two types. Adding QOC to already existing SOAP node creates QOC graph with an "empty" question (literal node) and one option corresponding to the SOAP node concept. The user should fill the question with the proper label (or comment) and probably add other options. VS currently does not support fuzzy weights at option nodes, but this could be part of future extension.
All three mentioned QOC types are taking into account mainly "Question" and "Option" nodes and their relation to owner SOAP node. Using "Criteria" nodes is the same for all three types and it does not affect SOAP node.

Limitations:

For *type 2* it seems to be reasonable to create "Options" automatically based on existing concept subtypes taken from given ontology. Unfortunately *Nautilus ontology* has a very flat structure and for most of concepts it gives too many subtypes (if any) so that the QOC graph will be totally overfilled with option nodes. Therefore automatic preparation of Options is not implemented. This feature could be useful in other domain or with different ontology with a deeper hierarchy. Virtual Staff also allows to have only one QOC for one SOAP node. It is not possible to have for example for the same node initial QOC graph (type 1) and another one for finding correct concept subtype (type 2) etc.

3 User Interface

3.1 Virtual Staff UI overview

From the user point of view, Virtual Staff is a visualization tool of patient case and graphical editor of SOAP and QOC graphs, with possible connection of nodes and arcs with the ontology (through Corese search engine) and data interchange with Nautilus DB of EPR(s) (through XML exchange format). In the Virtual Staff, we combine both models, described in previous chapter: SOAP to visualize the medical record and QOC in phase of decision to choose between pathologies or between action plans. Each patient case consists of exactly one SOAP graph and any number of QOC graphs. Any *A* or *P* node in SOAP graph could have an associated QOC graph. In Virtual Staff we distinguish between opened and closed QOC graphs. An opened QOC graph corresponds to a support for choosing a new decision, while a closed QOC graph means that a decision was already taken.

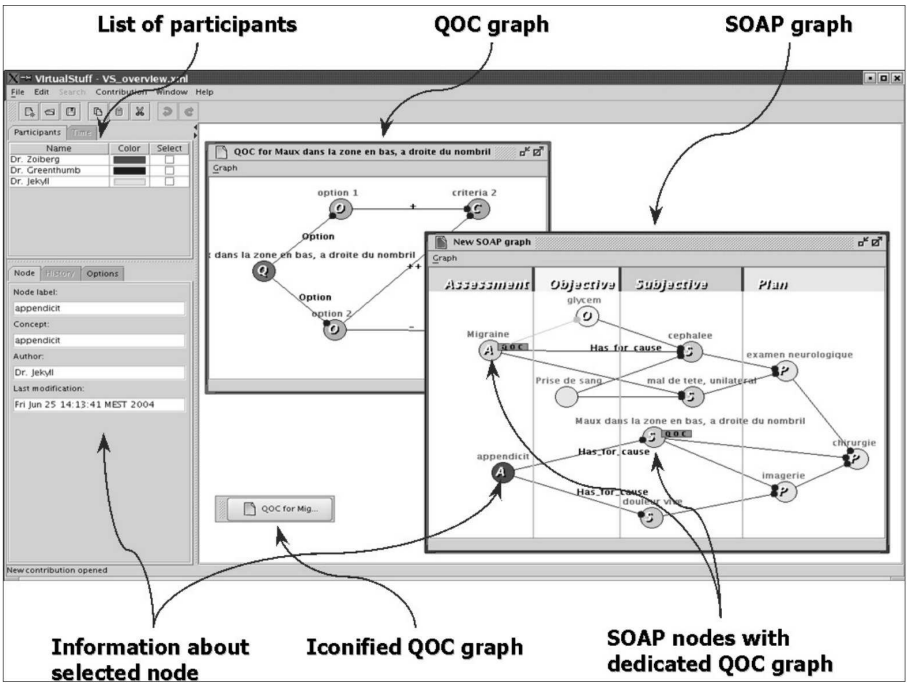


Figure 6. Virtual Staff UI overview

Fig. 6 gives a general overview of Virtual Staff user interface. Main (right) part of window represents electronic board for creating and updating graphs. Left part gives us information about selected graph and node/arc, about all involved participants and about history of the patient case.

It is possible to represent SOAP and QOC graphs through conceptual graphs, built by using the concepts and relations of the Nautilus ontology. Due to the correspondence between conceptual graphs and RDF(S) language, they can also be represented in RDF(S). Virtual Staff therefore implements save function using RDF(S) format.

Let us note that the Virtual Staff is not at all an expert system. It only allows the health care team to visualize the reasoning and the decision-making process; the reasoning is carried out by

the members of the network and the inferences based on the ontology enable the system to filter the choices offered to the user. The role of the QOC graph can be compared to a guide of best practices. But the QOC graph relies on the specific data of the concerned patient and not on generic data. To the nodes or arcs of the SOAP and QOC graphs, one could associate medical documents such as guides of best practices: an argument on the choice of a given treatment could be connected by a hypertext link to a guide describing the criteria of selection of this treatment.

SOAP Graph

Each case has exactly one SOAP graph. It would not make sense to have several QOC graphs without general SOAP and it is hard to imagine situations which require several SOAP graphs for one patient at the same time. The SOAP graph is divided into four zones according to the SOAP model categories (*Subjective*, *Objective*, *Assessment*, *Plan*). The advantage of this approach is mainly in its clarity. Despite the fact that all nodes from same SOAP category have its particular colour, without clustering them into specified zones the graph becomes soon chaotic. All zones could be easily resized and their position is also saved in the data file.

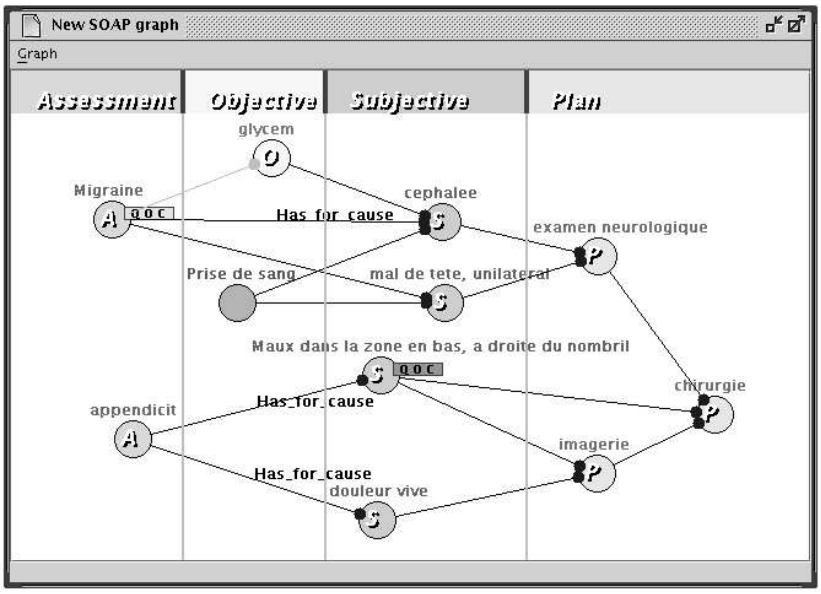


Figure 7. Example of SOAP graph

After opening a new case, VS automatically creates a new SOAP graph. Handling the SOAP graph is very simple. Almost everything is done by pop-up menus which appear after pressing the right mouse button. A content of the particular pop-up menu depends on a place where the user clicks the mouse button (e.g. on an already existing node, pop-up menu contains options like *edit node/remove node/add relation,* on a blank part of the graph there is option *add new node*s etc.). Nodes can be dragged by mouse (left button) and moved to any part of the graph. Dragging one node to another using right mouse button adds a new relation between these nodes. Both SOAP and QOC graphs are oriented, so while adding a new relation, first selected node is the *father node* and the next one is the *son node*. This is important for relation types taken from ontology which are also oriented.

Importing data from Nautilus database

When a new instance of Virtual Staff is created, the user has two possibilities – to start with an "empty graph" or to import data from Nautilus database. In the graph there has to be way how to distinguish between *common nodes* and *imported nodes*, so imported nodes are decorated with "black corners" (see fig. 8). No node imported from Episodus can be deleted nor its concept modified.

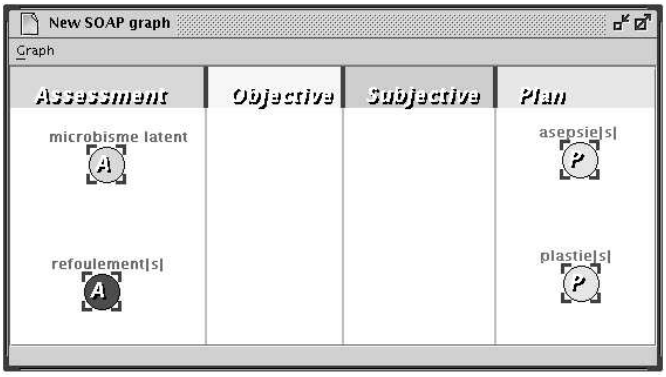


Figure 8. Example of imported data

3.2 QOC Graph

In general, QOC graph is quite similar to the SOAP graph. Handling nodes and relations in QOC graphs is the same as for SOAP graphs, only node types (*Question*, *Option*, *Criteria*) and relation types are different. Of course QOC nodes cannot be assigned with a link to other QOC graphs.

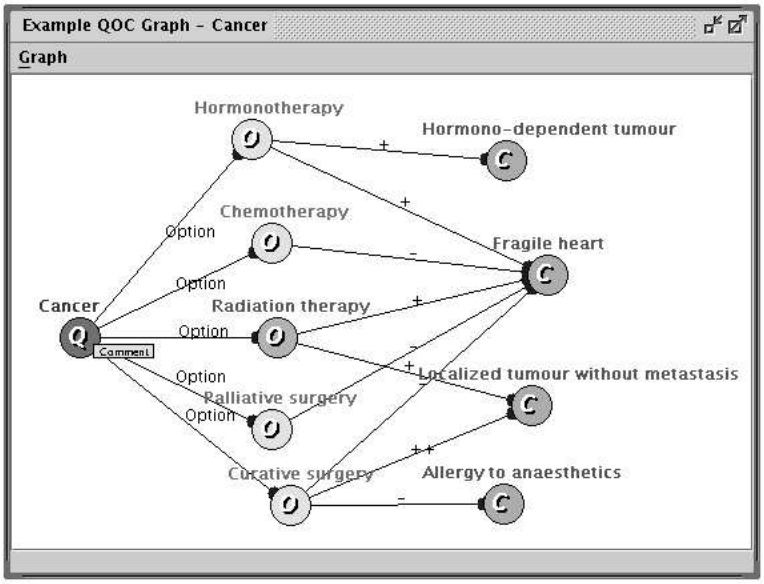


Figure 9. Example of QOC graph

Between *Option* and *Criteria* nodes could have positive or negative relation. Instead of fuzzy weights at relations, Virtual Staff is currently using only "normal" and "strong" relations. This

leads to four relation types - strong positive relation (with label "++"), normal positive ("+"), normal negative ("-") and strong negative ("--") (see fig. 9).

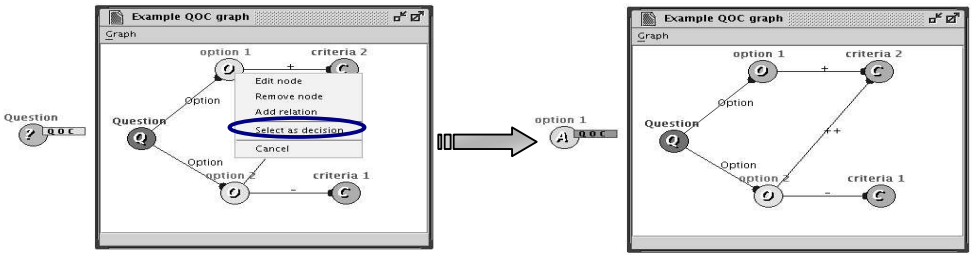
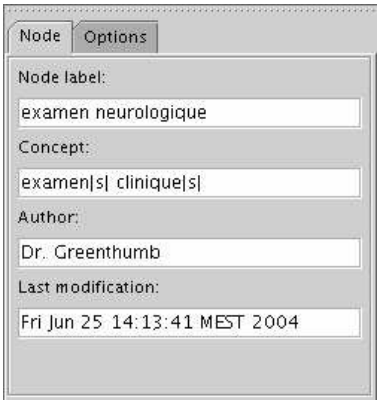


Figure 10. Selecting QOC decision

Important feature about QOC graphs is picking their decisions. Every “option” in QOC graph has as default yellow colour. When the user selects any option as decision (using pop-up menu and clicking on “*Select as decision*”), its colour changes to green. SOAP node connected with QOC graph has special “QOC” label with specific colour indicating QOC state (yellow colour for undecided QOC graph and green colour decided QOC). Changes on the SOAP node and the QOC graph are shown on fig. 10.

Node property dialog



From the user point of view each node has four attributes:

- Nautilus concept type
- Label
- Author (leading participant of the contribution)
- Date of last modification

For picking of Nautilus concept type the user selects one of nine major types and then simply types starting letters in *"Select concept name"* field. VS automatically propose list of existing concept types from selected category. When no concept type (or invalid one) is selected, node becomes *literal node*. Such node could be assigned with correct concept type later. When the user omits to fill *label,* VS uses standard label from ontology.

Note that *author* and *date of last modification* are handled automatically and common user cannot alter them.

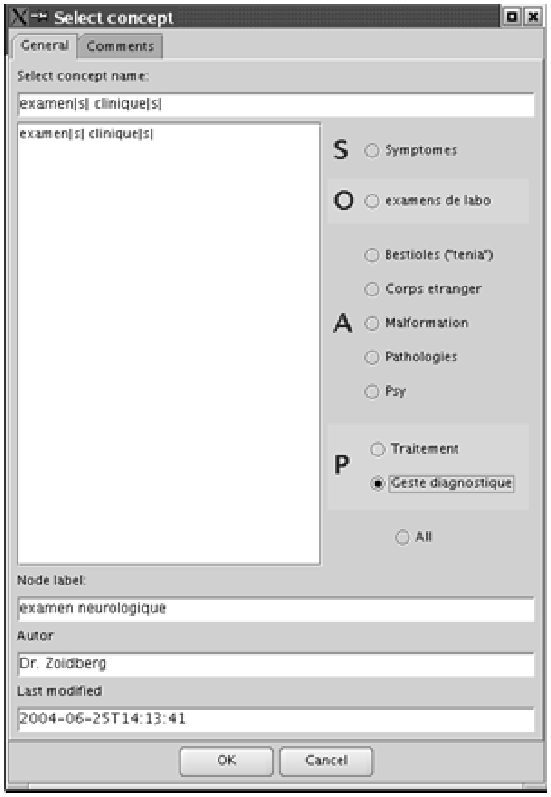
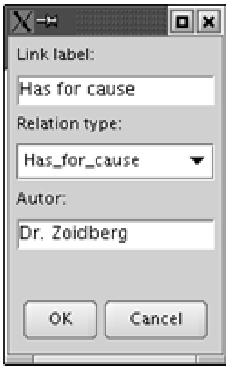


Figure 11. Node information

Relation property dialog:

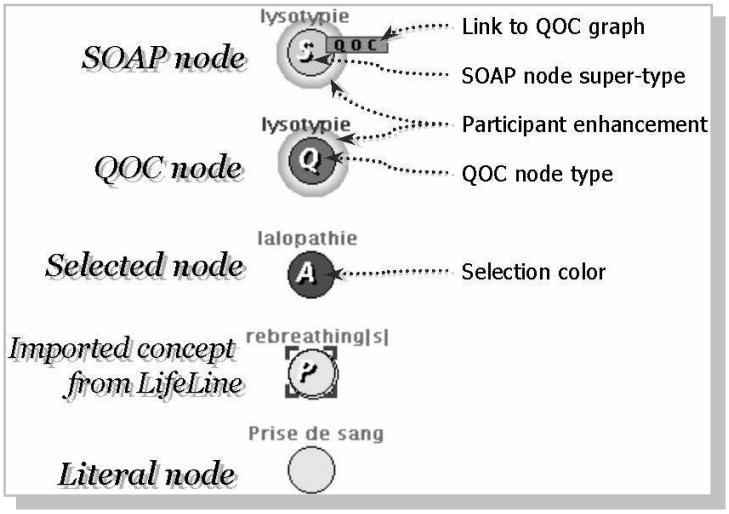


Relation property dialog is almost the same as the previous one. The only difference is in *Relation type* attribute, which replaces nodes' *concept type*.

Node GUI

It was necessary to make graphs comprehensible at the first sign. The most important thing was to put all significant information within graphical image of the node. In general, each node consists of a body and a label. The label gets default value from the *Nautilus ontology* or it is typed by the user. Inside node body there is a character indicating the SOAP category or QOC node type. Both SOAP and QOC graphs have also their own colour schema. SOAP nodes with dedicated QOC graph are displayed with a small **QOC** label.

Sometimes nodes can be enhanced with a colour ring according to their author (the participant who creates them). Each author has unique colour, so it is easy to see who made which part of the graph.

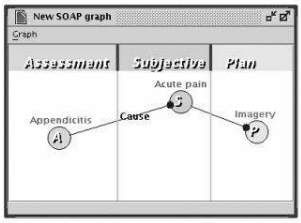


When the user clicks mouse on any node, detailed information about it is displayed in bottom left part of the main window. Selected node is filled with red colour. Finally, *literal nodes* (nodes which are not connected to any concept from *Nautilus ontology*) have light grey colour and no special character inside body.

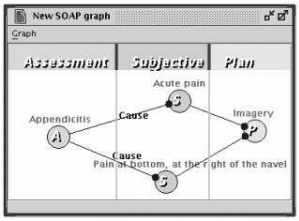
3.3 Sessions and Contributions

One of our assumptions is that any change in the graph must have a responsible participant. Each time a participant (any doctor or other medical person) opens new or already existing case, Virtual Staff automatically creates new *"contribution"* for this participant. Everything performed during this contribution (adding/changing nodes and relations) is stored in an XML file using *delta function* and for tractability reasons, it cannot be directly altered later. For example one doctor adds couple of nodes in SOAP graph and then closes his/her contribution. Then another doctor in a following contribution can modify these nodes and/or delete them, but Virtual Staff will remember both states, before and after modification. On the other hand, Virtual Staff cannot manage changes made within one contribution (e.g. when a doctor adds a new node and then removes it, nothing is stored in the delta function; if (s)he altered the name of some node several times, only the last value is stored, etc.).

Fig. 12. presents two states of SOAP graph made by two consecutive contributions and shows the corresponding parts of XML file (for saving space we skip definition of relations between nodes). In first part is defined an initial state of SOAP graph, this means that every node and relation is stored. Whereas in second part there are only new nodes and modifications to existing nodes. This approach has several advantages. The user can easily find who is responsible for any part of any graph at any time, can backtrack to any previous contribution and see evolution of the patient's case and thanks to delta function, (s)he can also see whether some parts of graph were deleted and/or changed by different participants.




```
<Contribution open="2004-07-26T17:00:02" close="2004-07-26T17:04:44" participant="Dr. Zoidberg">
  <SOAPGraph title="New SOAP graph">
    <NodeList>
      <Node ID="n0" label="Appendicitis" x="67" y="124">
        <Concept ID="PAPPA1"/>
      </Node>
      <Node ID="n1" label="Acute pain" x="229" y="76">
        <Concept ID="SD0UL1"/>
      </Node>
      <Node ID="n2" label="Imagery" x="330" y="116">
        <Concept ID="GIMAG1"/>
      </Node>
    </NodeList>
    <RelationList> <!--  skipped, similar as Nodes --> </RelationList>
  </SOAPGraph>
</Contribution>
```




```
<Contribution open="2004-07-26T17:00:02" close="2004-07-26T17:06:49" participant="Dr. Jekyll">
  <SOAPGraph>
    <NodeList>
      <Node ID="n1">
        <Concept ID="SD0311"/>
      </Node>
      <Node ID="n3" label="Pain at bottom, at the right of the navel" x="211" y="184">
        <Concept ID="SD0UL1"/>
      </Node>
    </NodeList>
    <RelationList> <!--  skipped, similar as Nodes --> </RelationList>
  </SOAPGraph>
</Contribution>
```

Figure 12. XML format for storing cases using delta function

The user does not take care about opening and closing sessions. The session is opened when a saved case is loaded or a new one is created. Inside this session user(s) (participants) can open any number of contributions. The session is closed automatically each time before leaving the current case. Aside from *open* and *close contribution* functions VS has also a *reset contribution* function. This function restores all graphs to their states as if no modification was made during the last contribution. It is also possible to reset the contribution only for one specific graph.

Review mode :
Virtual Staff supports "review mode" . Using this mode VS can easily backtrack to any previous state of graph stored in delta function in the patient case. Unit for backtracking is one contribution. So it is possible to see changes made during one contribution by one participant. With respect to this fact users should sometimes separate session in several contributions to propose more understandable patient case in review mode. Using buttons the user can step forward or backward through the patient case history.

Figure 13. shows simple example with two closed contributions. Left side represents state after first contribution (with three nodes only) and right side after second one (full patient case). Nodes and arcs displayed at review mode cannot be moved or updated, but the user can still use graphical enhancements (e.g. participant enhancement).

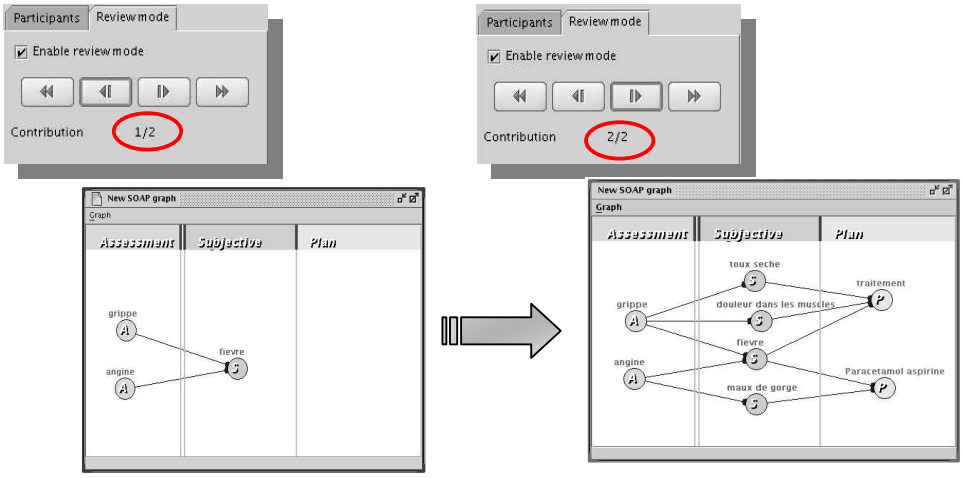


Figure 13. Process of review mode

3.4 Other features

Drag and drop

It seems feasible that nodes with the same concept and description will be used in several graphs at the same time. To retain VS as user friendly as possible it is much more effective to enable *drag and drop* functions rather than creating same node each time manually. Another reason for Drag and Drop is an expected connection between VS and the Episodus. *Drag and drop* support in VS is therefore designed to be able to transfer data with other native applications.

Modifications in graph

Each graph has a *document* icon in upper left corner. To have better overview of which graphs were modified and which not, this icon differs according to performed modifications in the graph. It is possible to acquire previous state of the graph by resetting last contribution. Also after saving case, all graphs are marked as unmodified.

Figure 14. Modified graph (on left) and untouched graph (on right)

Participants highlighting

Some patient cases are created by more than one doctor (participant). In the top left corner of the main window is a list of all participants involved in the current graph, where each participant has its own colour and check-box. Selection of any check-box highlights nodes created by the corresponding participant in all graphs.

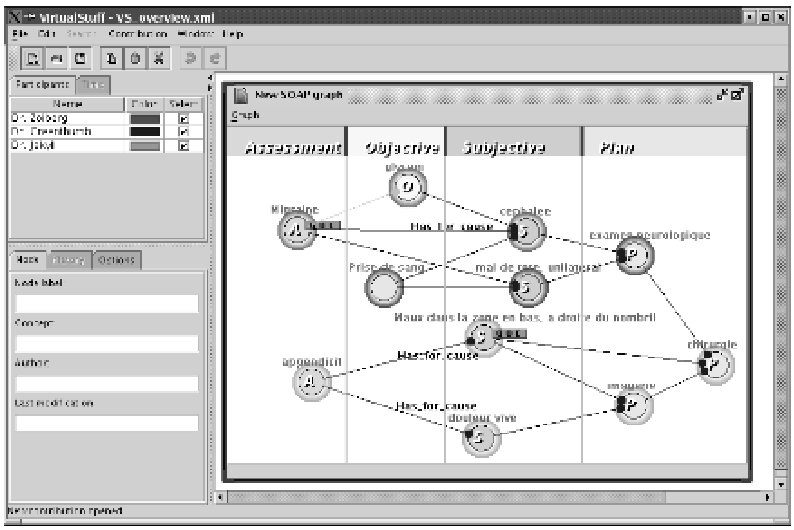


Figure 15. Participants highlighting

Selection of particular participant/contribution will enhance appropriate nodes with *colour rings.* These *colour rings* depend on the contribution author, its role or the last modification time. This enables quick overview of which part of graph was created by which participant. Colour enhancement could also help new participant to better and faster understand patient case. After brief examination of single case using these enhancements and *review mode*, the doctor should have sufficient information about case evolution and about reasoning of other participants.

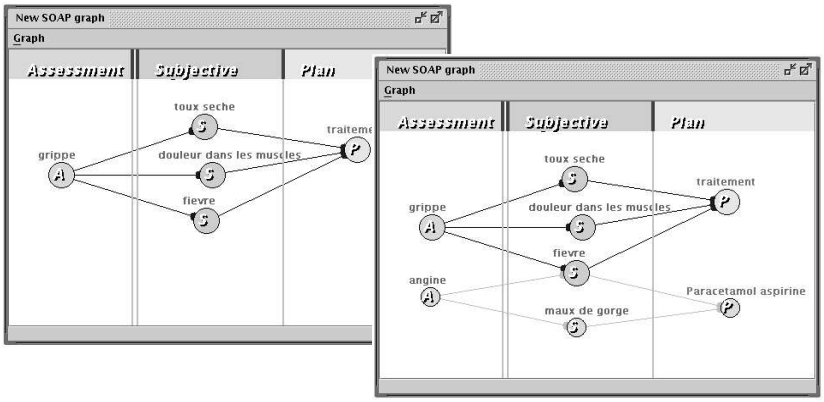


Figure 16. Showing removed nodes and relations

Graph visualization options

There are currently two options with influence on a visual aspect of graphs. First one is *Show removed nodes and relations*. Because all graphs are built using the delta function, even deleted nodes and relations are still in the *VS memory*. Usually deleted items are displayed in light grey colour and deleted nodes have smaller body. When the user wants to see the graph with active items only, s(he) can switch this option off (see fig. 16).

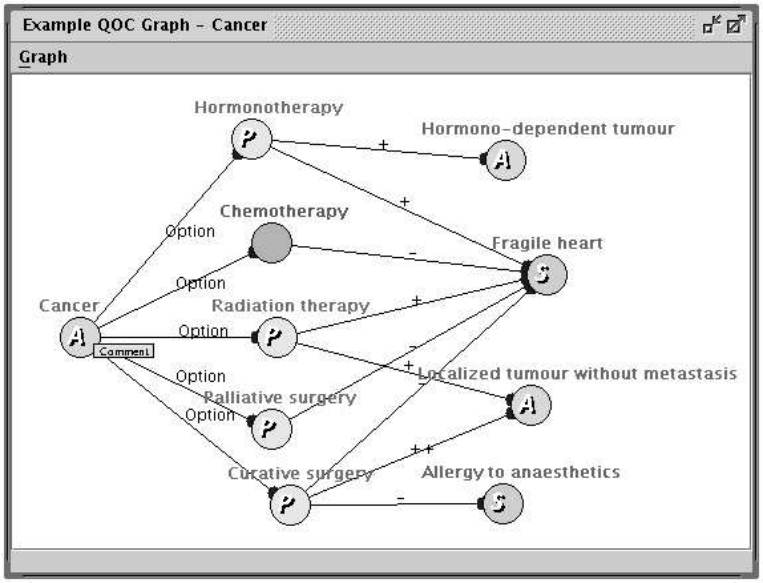


Figure 17. Showing SOAP category labels on QOC graph

Sometimes the user may want to see labels of SOAP categories on QOC graph. It is possible to switch QOC graph visualisation to "SOAP" style. Such graph is shown on fig. 17. (compare with fig. 9 where it is displayed as original QOC graph)

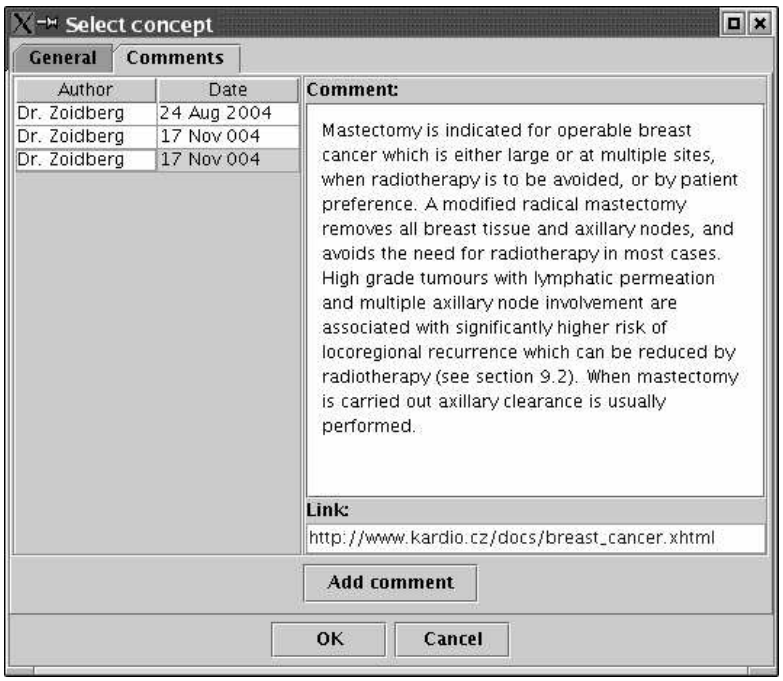


Figure 18. Example of comment

Comments and references

Another important feature is adding of comments to nodes. This could be valuable especially during reasoning in QOC graphs. Comments are added through the "Comments" tab in node property dialog. Each comment has its author (participant) and date, and it consists of textual comment or URL link (or combination of both).

As well as SOAP nodes with dedicated QOC graphs have special **QOC** labels, nodes with comments have **Comment** labels. With these labels the user can easily see which nodes are commented and which are not.

4 Cooperative work with Virtual Staff

Some patient cases are solved by a single doctor, but other complex cases need cooperation of several doctors having different medical specialties (in particular, in networks dedicated to heavy pathologies). Cooperation itself could be divided by time in *synchronous* and *asynchronous* and by place in *face-to-face* and *distant*. In ideal world, all cases will be solved in synchronous face-to-face sessions, but in practice doctors have to overcome "gaps" between places and time. While designing Virtual Staff, we had to take into account multi-users aspect with respect to all possible cooperation types.

Today it is easy to share XML or RDF(S) files, so the problem of distance between places is not the crucial one. No software tool can be sophisticated enough to fully replace natural collective discussion, but especially in medical domain, where there are many specialists and competence fields, it is difficult to get all required doctors in a same place at the same time. In Virtual Staff, it is possible to edit a patient's case asynchronously and members of health network can append their part of "knowledge" in the case at any time. Moreover Virtual Staff provides new users with valuable information about patient's case history.

It is understandable that each specialist could have a little bit different opinion on the same patient's case. After the patient's visit at a general practitioner, the doctor could for example prepare a QOC graph for some new symptoms to diagnose. From his/her point of view (s)he makes general model of decision process using this QOC graph according to his/her medical specialty. Then if another doctor e.g. cardiologist, examines the graph from viewpoint of his/her medical specialty, the graph could be enriched with new options and criteria and possibly with new decision of QOC. This process could continue iteratively for other relevant medical specialties.

If every participant is obliged to introduce his/her medical specialty on the VS (we call it *role of participant*) and as far as all graph states and decisions are saved using delta function, knowledge contained in graphs can be analysed from much more complex view. Searching history of patient's case enables to see not only patient state for a given time, but it also brings us information about reasoning of all participants. In QOC graphs, the user can find out answers to questions like "*why and when particular treatment started*", "*which criteria were taken in account when treatment started*" etc. Each doctor could see which diseases and health care procedures were active during decision making and so on. If treatment failed for some reason and should be replaced (through a new QOC decision), then it is possible to list through all previous treatments and situations when these treatments were applied, and then to decide new treatment or to continue the older one (e.g. if new criteria recommending this treatment are given). So, Virtual Staff helps to visualise progress of all contributions and clearly shows reasoning of different participants.

If we implement save function exporting all graph states in separate RDF(S), we will get larger case repository containing every case multiple times for each finished contribution. This allows us to extend Corese queries with time and participant role aspect. Unfortunately this may lead to huge RDF(S) archive but it will be used only in special cases for general analysis or data-mining, whereas repository with only final states of cases will be used for common case-based reasoning.

5 Implementation of Virtual Staff

5.1 User actions

Patient case

New case Select "*File/New case*" from menu bar. If the previous patient case was not saved, the user is asked whether s(he) wants to save modifications. New case also starts automatically new session and opens new *contribution dialog* (see *Session/Contribution* section below). Cases based on data acquired from the Episodus tool are not initiated using this option. Use *Open existing case* instead.

Open existing case Select "*File/Open*" from menu bar. Use this option for opening both saved patient cases and data files acquired from the Episodus tool. After opening existing case, VS starts automatically new session and opens new *contribution dialog* (see *Session/Contribution* section below).

Save case For saving patient case select "*File/Save*" from menu bar. It is possible to save case in file with a different name using "*File/Save As*". Both options cause closing of current session and contribution and re-opening new ones automatically with the same participant.

Export case to RDF Select "*File/Save to RDF*" from menu bar. The patient case is saved in file with the same name and path as XML patient file with extension *.rdf.

Contribution/Session

Open/close session Opening and closing sessions is provided fully automatically by VS. Session is opened while creating new case or opening existing one and closed whenever patient case is saved in XML file.

Open contribution Select "*Contribution/Open contribution*" from menu bar. Use this option each time, when you want to divide current session into several contributions (e.g. new leading participant). Opening contribution shows new *contribution dialog*, where it is necessary to specify new leading participant (select name from existing list or type new one). *Contribution dialog* is launched while creating new patient case or opening existing one.

Close contribution Select "*Contribution/Close contribution* " from menu bar. Use this option each time, when you want to add new contribution to current session with the same leading participant.

Reset contribution Select "*Contribution/Reset contribution*" from menu bar. All user actions made during last contribution are cancelled and all graphs are restored into their previous states. Resetting contribution does not have influence on current contribution and session attributes (opening/closing times and leading participant are still the same). Once closed contribution cannot be reset.

Handling graphs

Add new node Press right mouse button on blank part of the graph and select *"Add new node*" option from pop-up menu. New *node property dialog* appears. After confirming this dialog, new node is added in graph.

Edit node Press right mouse button on existing node and select *"Edit node*" option from pop-up menu. *Node property dialog* for selected node appears.

Remove node Press right mouse button on existing node and select *"Remove node*" option from pop-up menu. If the node was added into graph during current contribution, it is removed from graph. Otherwise the node is marked as "inactive" graph item. (see *graph visualization options* section below).

Moving nodes Drag node using left mouse button and drop it at any position on graph. It is also possible to drag/drop from one graph to another.

Add new relation Press right mouse button on existing node (father node) and release it on another node (son node). New *relation property dialog* appears. After confirming this dialog, new relation is added in graph.

Edit relation Similar to "*Edit node*".

Remove relation Similar to "*Remove node*".

Reactivate node/relation

Deleted nodes and relations marked as inactive graph items could be "reactivated" to normal state by selecting "*Reactivate*" option from pop-up menu.

Creating new QOC graphs

Press right mouse button on existing node and select *"Edit QOC*" option from pop-up menu and than fill necessary information in *new graph dialog*. This will create new QOC graph connected to selected node. It is possible to add QOC graph also to node that does not exist yet. Press right mouse button on blank part of the graph and select *"Add new node with QOC link*". Creating new QOC graphs is available only for SOAP nodes.

Participants highlighting

When the user activates any graph, list of all involved participants is prepared in the "*Participants*" panel in top left area of main window. Every participant has its particular colour, which could be changed. To activate participant highlights select checkbox in his/her row. All nodes made by this participant will be then highlighted by special colour rings.

Review mode

Starting review node To start review mode point to "*Review mode*" panel in top left area of the main window and select "*Enable review mode*" checkbox. Now use arrow buttons to navigate through previous contributions. While using *review mode* it is impossible to make any changes in graphs, but it is still possible to use graphical highlights.

Graph visualization options

"*Graph visualization options*" are listed at "*Options*" panel in bottom left area of main window.

Show removed nodes and relations

This option enables to see or hide inactive graph items (deleted nodes and relations).

Show SOAP categories on QOC graph

This option displays SOAP labels (e.g. "S") even on QOC graph nodes according to their concepts.

5.2 Data formats

A crucial thing for following development of Virtual Staff (VS) was to define data format for storing all necessary information. There are several requirements for this format. Most important one is that we want to store not even all graphs as they are but also detailed history, how they were built. It would not be effective to store whole state of graphs after each use of VS, hence data format should support "delta function" to be able to store initial state of graphs and then only changes made during last session.
To enable synchronicity and multi-user we need to define *Session* and *Contribution*. Each time VS opens some case for editing will be called *Session*. Inside this session, since there is a single "pen" that allows editing (even for multi-user), each time someone takes the pen it opens a *Contribution*. Both session and contribution need appropriate attributes:

Session :

- opening date and time
- closing date and time
- list of participants (some may never contribute ; however they were here)

Contribution :

- opening date and time
- closing date and time
- user (active participant)

Here is example of *session* and *contribution* written in XML (Note: list of participants will be described later. Time and date is encoded in XML Schema DateTime data type).

```
<Session openDate="2004-05-30T09:00:00" closingDate="2004-05-
30T10:00:00">
   <Contribution participantID="Dr. Greenthumb"
         openDate="2004-05-30T09:00:00" closingDate="2004-05-
30T09:20:00">

      ... Contribution contend ...
   </Contribution>
</Session>
```

Contribution can hold new SOAP and QOC graphs (initial states) and/or changes to already existing graphs. Each graph comprises nodes and their relations. Therefore we define elements `NodeList` (with sub-elements `Node`) and `RelationList` (with sub-element `Relation`). Next example shows simple SOAP graph initial state definition.

```
<SOAPGraph title="example of SOAP graph">
```

```
   <NodeList>
      <Node ID="n1" label="new node" x="100" y="100">
         <Concept ID="TFROV1" />
      </Node>
      <Node ID="n2" label="acide ascorbique" x="100" y="200">
         <Concept ID="TT1A41" />
         <QOCGraphLink ID="qoc_01" />
      </Node>
   </NodeList>
   <RelationList>
      <Relation  ID="r1"  type="has-for-cause"  label="has  for
cause">
         <FromNode ID="n1" />
         <ToNode ID="n2" />
      </Relation>
   </RelationList>
</SOAPGraph>
```

From example we can see that each node has its unique identifier, label, coordinates and concept referencing to Nautilus ontology (in case that node is not a literal). If node is connected with QOC graph it has additional sub-element QOCGraphLink with attribute referencing to appropriate QOC graph ID. Relations have also unique identifier and label. They could have optional attribute type which corresponds to one of predefined relation types encoded in RDF(S). Each relation must define `FromNode` and `ToNode` sub-elements.
There is no essential difference between `SOAPGraph` and `QOCGraph` definition. `QOCGraph` has in addition ID attribute (there is only one SOAP graph for the whole case, so it doesn't need ID) and sub-element `Decision` used in situations when QOC is already decided.

```
<QOCGraph ID="qoc_01">
   <NodeList> ... NodeList content ...
   </NodeList>
   <RelationList>
      <Relation ID="qoc_01_r2" type="positive criteria"
weight="+">
        <FromNode ID="qoc_01_n2" />
        <ToNode ID="qoc_01_n3" />
      </Relation>
   </RelationList>
   <Decision node="qoc_01_n2" />
</QOCGraph>
```

Updating nodes and relations in graphs (*delta function*) is very simple. For demonstration let's start from previous `SOAPGraph` element as initial state of graph. If someone wants to change for example coordinates of one node (in our case node with ID="n1"), s(he) just puts `SOAP-Graph` element in new contribution, specifies `Node` sub-element with appropriate ID value and new coordinates values. All other attributes and sub-elements are omitted and `Node` will keep their values from initial graph state. Next example shows also how to change label and concept

(node "n2"). When user wants to remove node from graph, empty sub-element `Removed` is used inside `Node` element (node "n3"). Relations are updated in same way.

```
<SOAPGraph>
   <NodeList>
      <Node ID="n1" x="100" y="100" />
      <Node ID="n2" label="new label">
         <Concept ID="PBTKI" />
      </Node>
      <Node ID="n3">
         <Removed />
      </Node>
   </NodeList>
</SOAPGraph>
```

To provide connection to *LifeLine,* `Contribution` element could contain also `LifeLineOutput` sub-element. `LifeLineOutput` consists of all relevant concepts used in *LifeLine*. This element is not created within VS but it is completely imported from *LifeLine* interface. It means that rather than opening first session in VS, *LifeLine* automatically generates this first session with contribution containing initial `LifeLineOutput`. Inside this element there are several `ConceptIn` definitions referencing to Nautilus ontology concepts. For each `ConceptIn` VS creates label from Nautilus ontology and coordinates according to concept supertype ("Pathology", "Laboratory test",…) .

```
<Contribution participantID="Dr. Greenthumb" openDate="2004-05-
30T09:00:00" closingDate="2004-05-30T09:00:00">
   <LifeLineOutput>
      <ConceptIn nodeID="LL_01">
         <Concept ID="PHRAD1" />
      </ConceptIn>
      <ConceptIn nodeID="LL_02">
         <Concept ID="PBRGA1" />
      </ConceptIn>
   </LifeLineOutput>
</Contribution>
```

In first example element `Contribution` uses attribute *ParticipantID*. Value of this attribute is referencing to one of present participants. Every session must have some participants and their list is placed in obligatory sub-element `ParticipantPresent`. To enable storing information about each participant, there is more general element ParticipantList. Through this element new participants involved in case are added to or removed from list. Attributes of element AddParticipants will be altered later according to data available from *LifeLine* database.

```
<ParticipantsList>
    <AddParticipant ID="Dr. Zoiberg" role=" Medecin generaliste
/>
    <AddParticipant ID="Dr. Greenthumb" role=" Medecin generali-
ste " />
   <RemoveParticipant ID="Dr. Jekyll" />
```

```
</ParticipantsList>
<ParticipantsPresent>
   <Participant ID="Dr. Greenthumb" />
   <Participant ID="Dr. Zoiberg" />
</ParticipantsPresent>
```

Usage of participant list and all other mentioned elements and their attributes is also described in complex example[5] of VS XML data file.

[5] See Appendix A

6 Conclusions

Virtual Staff enables members of a health care network to create, modify, consult alone or together:

- SOAP graphs describing the links between diagnostic and therapeutic hypotheses, symptoms and observations,
- and QOC graphs for support to decision-making.

The nodes of both kinds of graphs are typed by the concepts of the Nautilus ontology. Such a combination of these SOAP and QOC models with an ontology is original and illustrates the interest of an ontology to help the user to visualize a reasoning or a decision-making process.
Virtual Staff is also designed for multi-user cooperative work with possible backtracking through evolution of cases. This enables to see how knowledge contained in graphs was modelled and how reasoning and decision making were processed. Solved and closed cases are stored in RDF(S) repository and they can be analysed to get new knowledge (to improve ontology etc.) or reused in case-based reasoning on new cases.
To reach platform-independence Virtual Staff is completely implemented in JAVA. Connection to Nautilus DB or other database of EPR(s) is optional, but to run Virtual Staff, the user needs to rely on an ontology represented in RDF(S).
The first version of the Virtual Staff was validated by Nautilus SARL, from two viewpoints: its functions and its interfaces. This evaluation led to several improvements of the graphical interface of the virtual staff: the new interface helps the user to really follow the reasoning guided by the SOAP model (each part of the graph is dedicated to nodes expressing either an Assessment or an Objective or a Subjective or a Plan) or by the QOC model (Question → Option → Criterion).
The Nautilus ontology and the Virtual Staff were validated by Nautilus society. Notice that, even though the Virtual Staff was implemented with the Nautilus ontology (for collaboration reasons), it would be possible to adapt the Virtual Staff to another medical ontology such as UMLS meta-thesaurus (Pisanelli et al, 1998).
As IBIS method is close to QOC method, we can compare the virtual staff to gIBIS, that already relies on graph visualization, and offers argumentation on decisions (Conklin and Begeman, 1988). But our originality is to combine both SOAP graphs and QOC graphs and to rely on a medical ontology for building and handling these graphs. Another cooperative tool for a healthcare network is WebOnColl, a web-based medical collaborative environment relying on user profiles and virtual workspaces (Chronaki et al, 1997).

6.1 Discussion

From organizational viewpoint, the organization constituted by a health care network can be considered as *a virtual enterprise,* with a rather informal structure, and its members constitute a community gathered by a common objective (i.e. offer the best health care and follow-up for the patients), each member having also more specific objectives due to his/her profession (e.g. doctor vs nurse vs social worker) and to his/medical specialty. The kind of cooperation in this organization may also depend on the kind of network: some networks are dedicated to a heavy pathology (e.g. diabetes, oncology, etc), and will gather members from different professions and different specialties, while other networks will rather gather the same kind of health centres or of professionals (e.g. hospital professional network, liberal practitioner network, pharmacist network, nurse network, etc) and others will be dedicated to a type of patient (new born, old people). The kind of interactions will of course differ according to the type of network. Our approach, based on support to cooperative reasoning, seems useful for a healthcare network dedicated to a heavy pathology. In (Bardram and Sølvkjær, 1996), four types of collaborative aspects in clinical research are emphasized: communication, sharing different records and mate-

rial, planning of collaboration and collaborative problem solving: our approach tries to tackle at least patient record sharing and collaborative problem aspects.

From *cognitive viewpoint*, the members of the network build themselves a mental representation of the patient's case. The graphs handled in the Virtual Staff aim at enabling a participant to visualize partly this representation, to share it with other participants and to make it evolve through cooperation with other participants. SOAP model seems relevant for medical reasoning and QOC for representing diagnosis and therapeutic decisions since QOC model is known as useful for design rationale of a design project: patient's health care and follow-up can be considered as a therapeutic project to which some members of the health care network will part in.

From *technological viewpoint*, in addition to SOAP and QOC graphs, we mainly rely on semantic Web technologies:

- ontologies for representing the concepts shared by the network members,
- RDF(S) for representation of ontology and annotations on the patient's record,
- Corese semantic search engine for querying the ontology and RDF annotations (in particular on one or several virtual staffs concerning one or several patients),
- XML for interchange between the virtual staff and the Episodus software describing the life line of the patient.

Our approach can be seen as tackling knowledge management from several viewpoints:

- *Organizational memory*: we build a memory of a specific community constituted by the health care network members: these members may be individual or organisations (e.g. hospital, health centre). More precisely, we build a memory of health projects about patients: each life line of a patient is considered as the trace of a medical project, with events, phases, actors playing a role in this project. Tractability of the project decision rationale is tackled by QOC graphs.
- *Support to cooperative work:* our approach follows the suggestions of (Pratt et al, 2004) for taking inspiration of CSCW for heath care support. We try to offer, in a longer-term, shared understanding, informed participation and social creativity, as in the vision of (Fischer, 2000).
- *Case-Based reasoning:* even though we do not use classic Case-Based Reasoning techniques (Moussavi, 1999), querying through Corese on past patient cases and past VS sessions in order to have suggestions for a new patient case aims at the same objective as case-based reasoning.

7 Future work

As a further work, we plan several extensions and improvements of our tool. We will offer Virtual Staff to medical community for testing. Possible perspectives of this work are:

- *Enrich Nautilus ontology* according to concepts and relations recommended by doctors,
- *Implement user-friendly interfaces* for visual definition of Corese queries and for presentation of Corese results in a way understandable by health actors,
- Improve the *cooperation of multiple users through the Virtual Staff*,
- Improve selection and handling of participants,
- Picking participants from *participants database,*
- Solving access rights,
- Including participant role.
- Last, Nautilus SARL will proceed to end-user focused evaluation by physicians taking part in an actual network (probably in diabetes).

8 Acknowledgements

We thank very much Philippe Ameline, the author of the Nautilus database who launched the project "Ligne de Vie", who had the initial idea of the virtual staff and who interacted very fruitfully with us for specification and validation of our work on the ontology and on the virtual staff, Frédéric Corby and Olivier Corby for the reconstitution of the Nautilus ontology from the Nautilus database, Pr P. Degoulet, Dr. Dericco, Mrs. Labelle and Mrs D. Sauquet, for our fruitful discussions, and the "Ministère de la Jeunesse, de l'Éducation Nationale et de la Recherche" for funding the project "Ligne de Vie".

9 References

Bardram, J. E. and Sølvkjær, M. (1996): *Computer Supported Cooperative Work in Clinical Practice*. Proceedings of the 13th Int. Congress of the European Federation for Medical Informatics, Copenhaguen, Denmark, 1996, p. 853-857.

Benbasat, I., Dexter, A., Todd, P. (1986): *An Experimental Program Investigating Colour-Enhanced and Graphical Information Presentation: An Integration of the Findings*. Communications of the ACM, 29:1094-1105, 1986.

Chronaki, C.E., Katehakis, D.G., Zabulis, X.C., Tsiknakis, M., Orphanoudakis, S.C. (1997): *WebOnCOLL: medical collaboration in regional healthcare networks*. IEEE Transactions on Technology in Biomedicine, December 1997, vol 01, no 4, p 257-269.

Conklin, J., Begeman, M. (1988): *gIBIS: A Hypertext Tool for Exploratory Policy Discussion.* Communications of the ACM, 140-152, 1988.

Corby, O., Dieng, R., Hébert, C. (2000): *A Conceptual Graph Model for W3C Resource Description Framework*. In B. Ganter, G. W. Mineau eds, Conceptual Structures: Theory, Tools, and Applications, Proc. of ICCS'2000, Springer-Verlag, LNAI 1867, Darmstadt, Germany, August 13-17, 2000, p. 468-482.

Corby, O., Faron-Zucker, C.(2002): *Corese: A Corporate Semantic Web Engine.* In Proceedings of the International Workshop on Real World RDF and Semantic Web Applications, in conjunction with 11th International World Wide Web Conference 2002 Hawaii. May 2002. http://paul.rutgers.edu/~kashyap/workshop.html

Corby, O., Dieng-Kuntz, R., Faron-Zucker, C. (2004): *Querying the Semantic Web with the CORESE Search Engine.* In R. Lopez de Mantaras and L. Saitta eds, Proc. of the 16th European Conference on Artificial Intelligence (ECAI'2004), subconference PAIS'2004, Valencia, 22-27 August 2004, IOS Press, p. 705-709.

Dieng, R., Corby, O., Giboin, A., Ribière, M. (1999*): Methods and Tools for Corporate Knowledge Management*, special issue on Knowledge Management, vol. 51, pp. 567-598, 1999. Academic Press.

Dieng-Kuntz, R., Matta, N. (2002*): Knowledge Management and Organizational Memories*, Kluwer Academic Publishers, July 2002.

Dieng-Kuntz, R., Minier, D., Corby, F., Růžička, M., Corby, O., Alamarguy, L., Luong, P. (2004): *Medical Ontology and Virtual Staff for a Health Network*, In Enrico Motta and Nigel Shadboldt eds, Engineering Knowledge in the Age of the Semantic Web: Proceedings of the 14th International Conference, EKAW 2004, Whittlebury Hall, UK, October 5-8, 2004, Lecture Notes in Computer Science Publisher: Springer-Verlag Heidelberg, Volume 3257, ISBN: 3-540-23340-7, pp. 187 - 202.

Fischer, G. (2000): *Shared Understanding, Informed Participation and Social Creativity: Objectives for the Next Generation of Collaborative Systems*, In R. Dieng, A. Giboin, L. Karsenty, G. De Michelis, eds. , Designing Cooperative Systems: The Use of Theories and Models, Proc. of the 4th Int. Conf. on the Design of Coop. Syst. (COOP'2000), IOS Press, 2000, p. 3-16.

Gangemi, A., Pisanelli, D., Steve, G. (1999): *An Overview of the ONIONS Project: Applying Ontologies to the Integration of Medical Terminologies.* In Data and Knowledge Engineering, vol. 31, 1999.

Georg, G., Seroussi, B., Bouaud, J. (2003): *Dérivation d'une base de connaissances à partir d'une instance GEM d'un guide de bonnes pratiques médicales textuel*. In R. Dieng-Kuntz ed, Actes des 14èmes Journées Francophones sur l'Ingénierie des Connaissances (IC'2003), Laval, Presses Universitaires de Grenoble, Juillet 2003.

Golebiowska, J., Dieng-Kuntz, R., Corby, O., Mousseau, D. (2002): *SAMOVAR : using ontologies and text-mining for building an automobile project memory*. In R. Dieng-Kuntz and N. Matta eds, Knowledge Management and Organizational Memories, p. 89-102, Boston, July 2002.

Lassila, O., Swick R. R. (1999*): Resource Description Framework (RDF) Model and Syntax Specification.* W3C Recomm., 22 Feb. 1999, http://www.w3.org/tr/rec-rdf-syntax/

Maclean, A., Young, R., Bellotti, V., Moran T. (1991): *Questions, Options, and Criteria: Elements of a Design Rationale for User Interfaces.* Int. Journal of Human-Computer-Interaction, 6(3/4):201-250. 1991.

Moussavi, M. (1999*): A Case-Based Approach to Knowledge Management*. In Aha D.W. (Ed). Proc. of the AAAI'99 Workshop on "Exploring Synergies of Knowledge Management and Case-Based Reasoning". July 1999; Orlando, FL. AAAI Press Technical Report WS-99-10.

Pisanelli, D. M., Gangemi, A., Steve, G. (1998). *An Ontological Analysis of the UMLS Metathesaurus*. Journal of the American Medical Informatics Association, 5:810-814, 1998.

Pratt, W., Reddy, M., McDonald, D., Tarczy-Hornoch, P., and Gennari, J. (2004): *Incorporating Ideas from Computer-Supported Cooperative Work.* Journal of Biomedical Informatics, Volume 37 , Issue 2 , April 2004, p. 128 - 137.

Sowa, J. F. (1994): *Conceptual Structures: Information Processing In Mind and Machine, Addison-Wesley*, 1984.

Weed, L.D. (1971): *The Problem Oriented Record as a Basic Tool in Medical Education, Patient Care and Clinical Research*. Ann Clin Res 3(3):131-134. 1971.

10 Appendix A. Example of Virtual Staff data file

```
<?xml version='1.0' encoding="windows-1250"?>
<!DOCTYPE VSCase SYSTEM "vscase.dtd">
<VSCase ID="case_complex">

<!-- FIRST SESSION ##############################################################
       - define new list of participants
       - select presented participants
       - import LL data into VS
       - create graphs -->
  <Session openDate="2004-05-30T09:00:00" closingDate="2004-05-30T10:00:00">
    <ParticipantsList>
      <AddParticipant ID="Dr. Zoiberg" role="Medecin generaliste" />
      <AddParticipant ID="Dr. Greenthumb" role="Medecin generaliste" />
      <AddParticipant ID="Dr. Jekyll" role="Medecin generaliste" />
    </ParticipantsList>
    <ParticipantsPresented>
      <Participant ID="Dr. Greenthumb" />
      <Participant ID="Dr. Zoiberg" />
    </ParticipantsPresented>

<!-- Contribution 1 ~~~~~~~~~~~~~~
       - import LL concepts -->
    <Contribution participantID="Dr. Greenthumb" openDate="2004-05-30T09:00:00"
                  closingDate="2004-05-30T09:00:00">
      <LifeLineOutput>
        <ConceptIn nodeID="LL_01">
          <Concept ID="PHRAD1" />
        </ConceptIn>
        <ConceptIn nodeID="LL_02">
          <Concept ID="PBRGA1" />
        </ConceptIn>
      </LifeLineOutput>
    </Contribution>

<!-- Contribution 2 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
    <Contribution participantID="Dr. Greenthumb" openDate="2004-05-30T09:00:00"
                  closingDate="2004-05-30T10:00:00">
      <SOAPGraph title="example SOAP graph">
        <NodeList>
          <Node ID=" n1" label="new node" x="100" y="100">
            <Concept ID="TFROV1" />
            <Reference url="referenceURL">
              <Comment>comment about reference</Comment>
            </Reference>
          </Node>
          <Node ID=" n2" label="acide ascorbique dans le plasma" x="100"
y="200">
            <Concept ID="TT1A41" />
            <COQGraphLink ID="qoc_01" />
          </Node>
          <Node ID="n3" x="100" y="300">
             <COQGraphLink ID="qoc_02" />
          </Node>
        </NodeList>
        <RelationList>
          <Relation ID="r1" type="has-for-cause" weight="100">
            <FromNode ID=" n1" />
            <ToNode ID="n2" />
            <Reference url="referenceURL">
              <Comment>comment about reference</Comment>
            </Reference>
            <Reference url="another reference" />
          </Relation>
        </RelationList>
      </SOAPGraph>
      <QOCGraph ID="qoc_01">
        <NodeList>
```

```
          <Node ID="qoc_01_n1" label="node_label" type="question" x="100"
y="100">
            <Concept ID="TBACI1" />
          </Node>
          <Node ID="qoc_01_n2" label="node_label2" type="option" x="100"
y="200">
            <Concept ID="NSEDA1" />
          </Node>
          <Node ID="qoc_01_n3" label="node_label3" type="criteria" x="100"
y="300">
            <Concept ID="TBARR1" />
          </Node>
          <Node ID="qoc_01_n4" label="node_label4" type="criteria" x="200"
y="300">
            <Concept ID="TBARR1" />
          </Node>
        </NodeList>
        <RelationList>
          <Relation ID="qoc_01_r1" type="has-option">
            <FromNode ID="qoc_01_n1" />
            <ToNode ID="qoc_01_n2" />
          </Relation>
          <Relation ID="qoc_01_r2" type="positive criteria" weight="+">
            <FromNode ID="qoc_01_n2" />
            <ToNode ID="qoc_01_n3" />
          </Relation>
          <Relation ID="qoc_01_r3" type="negative criteria" weight="-">
            <FromNode ID="qoc_01_n2" />
            <ToNode ID="qoc_01_n4" />
          </Relation>
        </RelationList>
        <Decision node="qoc_01_n2" />
      </QOCGraph>
    </Contribution>
  </Session>

<!-- SECOND SESSION ##############################################################
       - add/remove participants
       - import new concept into VS
       - contribution for graph changes
       -->
  <Session openDate="2004-06-01T14:00:00" closingDate="2004-06-01T15:00:00">
    <ParticipantsList>
      <RemoveParticipant ID="Dr. Jekyll" />
      <AddParticipant ID="Mr. Hyde" role="consultant" />
    </ParticipantsList>
    <ParticipantsPresented>
      <Participant ID="Dr. Greenthumb" />
    </ParticipantsPresented>

<!-- Contribution 1 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
    <Contribution participantID="Dr. Greenthumb" openDate="2004-06-01T14:00:00"
                  closingDate="2004-06-01T14:00:00">
      <LifeLineOutput>
        <ConceptIn nodeID="LL_03">
          <Concept ID="PGHER1"/>
        </ConceptIn>
      </LifeLineOutput>
    </Contribution>

<!-- Contribution 2 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
    <Contribution participantID="Dr. Greenthumb" openDate="2004-06-01T14:00:00"
closingDate="2004-06-01T14:30:00">
      <QOCGraph ID="coq_01">
        <NodeList>
          <Node ID="coq_01_n4" x="120" y="150">
            <Concept ID="PHB001" />
          </Node>
          <Node ID="coq_01_n2">
            <Removed />
          </Node>
```

```
        </NodeList>
        <RelationList>
          <Relation ID="coq_01_r1">
            <ToNode ID="coq_01_n4" />
          </Relation>
        </RelationList>
        <Decision>
          <Removed />
        </Decision>
      </QOCGraph>
    </Contribution>

  </Session>

</VSCase>
```

11 Appendix B. VS Relation types encoded in RDF(S)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY cos   "http://www.inria.fr/acacia/corese#">
  <!ENTITY rdf   "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <!ENTITY rdfs  "http://www.w3.org/2000/01/rdf-schema#">
  <!ENTITY xsd   "http://www.w3.org/2001/XMLSchema#">
  <!ENTITY lv    "http://www.inria.fr/acacia/2003/lv#">  ]>

<rdf:RDF xmlns:rdfs="&rdfs;" xmlns:rdf="&rdf;"  xmlns:cos="&cos;"
xmlns:lv='&lv;'
xml:base='&lv;'>

<rdf:Property rdf:ID="SOAPRelation">
  <rdfs:domain rdf:resource="#ConceptNautilus"/>
  <rdfs:range  rdf:resource="#ConceptNautilus"/>
  <rdfs:subPropertyOf rdf:resource="#Relation"/>
</rdf:Property>

<rdf:Property rdf:ID="QOCRelation">
  <rdfs:domain rdf:resource="#ConceptNautilus"/>
  <rdfs:range  rdf:resource="#ConceptNautilus"/>
  <rdfs:subPropertyOf rdf:resource="#Relation"/>
</rdf:Property>

<rdf:Property rdf:ID="has_for_cause">
  <rdfs:domain rdf:resource="#S"/>
  <rdfs:range  rdf:resource="#P"/>
  <rdfs:subPropertyOf rdf:resource="#SOAPRelation"/>
  <rdfs:label xml:lang="en">has for cause</rdfs:label>
</rdf:Property>

<rdf:Property rdf:ID="has_for_consequence">
  <rdfs:domain rdf:resource="#P"/>
  <rdfs:range  rdf:resource="#S"/>
  <rdfs:subPropertyOf rdf:resource="#SOAPRelation"/>
  <rdfs:label xml:lang="en">has for consequence</rdfs:label>
</rdf:Property>

<rdf:Property rdf:ID="confirmed_by">
  <rdfs:domain rdf:resource="#P"/>
  <rdfs:range  rdf:resource="#T"/>
  <rdfs:subPropertyOf rdf:resource="#SOAPRelation"/>
  <rdfs:label xml:lang="en">confirmed by</rdfs:label>
</rdf:Property>

<rdf:Property rdf:ID="treated_by">
  <rdfs:domain rdf:resource="#S"/>
  <rdfs:domain rdf:resource="#P"/>
  <rdfs:range  rdf:resource="#N"/>
  <rdfs:subPropertyOf rdf:resource="#SOAPRelation"/>
  <rdfs:label xml:lang="en">treated by</rdfs:label>
</rdf:Property>

<rdf:Property rdf:ID="option">
  <rdfs:range  rdf:resource="#ConceptNautilus"/>
  <rdfs:domain rdf:resource="#ConceptNautilus"/>
  <rdfs:subPropertyOf rdf:resource="#QOCRelation"/>
  <rdfs:label xml:lang="fr"> </rdfs:label>
</rdf:Property>
```

```
<rdf:Property rdf:ID="positive_ criteria">
  <rdfs:range  rdf:resource="#ConceptNautilus"/>
  <rdfs:domain rdf:resource="#ConceptNautilus"/>
  <rdfs:subPropertyOf rdf:resource="#QOCRelation"/>
  <rdfs:label xml:lang="fr">+</rdfs:label>
</rdf:Property>

<rdf:Property rdf:ID="negative_ criteria">
  <rdfs:range  rdf:resource="#ConceptNautilus"/>
  <rdfs:domain rdf:resource="#ConceptNautilus"/>
  <rdfs:subPropertyOf rdf:resource="#QOCRelation"/>
  <rdfs:label xml:lang="fr">-</rdfs:label>
</rdf:Property>
</rdf:RDF>
```