Incremental guideline formalization with tool support

Radu Serban[1], Anna Puig-Centelles[2], Annette ten Teije [1]
[1] Department of Artificial Intelligence,
Vrije Universiteit, Amsterdam, The Netherlands
{serbanr,Annette}@cs.vu.nl
[2] Departament d' Enginyeria i Ci'encia dels Computadors,
Universitat Jaume I, Castellon, Spain
apuig@lsi.uji.es

Abstract. Guideline formalization is recognized as an important component in improving computerized guidelines, which in turn leads to better informedness, lower inter-practician variability and, ultimately, to higher quality healthcare. By means of a modeling exercise, we investigate the role of guideline formalization tools which use two different knowledge transformation principles in producing re-usable knowledge objects useful for representing medical processes and performing updates of medical guidelines. We give a general evaluation of usefulness and state the main requirements for tools that reuse medical knowledge and support authoring of guidelines.

1 Introduction

In recent years, medical guidelines and protocols have become the main instruments for disseminating best practices in clinical medicine. They promote safe practices, reduce inter-clinician practice variations and support decision-making in patient care while containing the costs, therefore leading to improvement of the quality and consistency of healthcare. Concerns for quality, consistency and uniformity of care stimulated the elaboration of a guideline development process for producing computerized evidence-based guidelines. Guideline formalization has been recognized as an essential component in this process, which leads to higher quality computerized guidelines. *Guidelines* set out what should happen, when, and by whom in the care process, and *guideline formalization* is the medical knowledge-driven transformation process by which an informal guideline text is translated into an equivalent but more formal representation, suitable for simulation and verification of properties. The resulting formal specification is tested with respect to requirements for medical processes, such as avoiding critical conditions, asserting the desired ordering of events, or describing how a desired situation can be reached.

Please use the following format when citing this chapter:
Serban, Radu, Puig-Centelles, Anna, Teije, Annette Ten, 2006, in IFIP International Federation for Information Processing, Volume 204, Artificial Intelligence Applications and Innovations, eds. Maglogiannis, I., Karpouzis, K., Bramer, M., (Boston: Springer), pp. 106–118

The potential flaws discovered result in criticism for the part of the guideline that generated the faulty specification.

Currently, the knowledge transformation employed for guideline formalization cannot be performed in one step, due to the complexity of the modeling knowledge required. Even worse, there are no rules for recognizing and transforming frequently encountered medical situations described in narrative medical text into their formal representation. Therefore, the knowledge engineers who perform the formalization cannot effectively apply such rules when similar guidelines are formalized, but rather have to rediscover them from scratch. Recent research on guideline formalization ([5, 6, 7, 10]) propose to solve this problem by making explicit and grouping together the different types of knowledge employed by the knowledge engineers in the formalization process and maintaining a mapping between the textual representation and the formal representation of the medical knowledge. This structuring of medical knowledge according to formal methods techniques ([15]) can then be used in the different phases of the, by now, standardized process of guideline formalization. Tracking of faulty fragments throughout the formalization phases enables guideline improvement.

There are very few tools that currently support a *step-wise knowledge refinement and knowledge tracking methodology for guideline formalization.* We selected two general purpose document management tools which employ different principles for structuring and transformation of knowledge, which makes them suitable for implementing this methodology: the Document Exploration and Linking Tool/Add-ons (DELT/A, formerly known as Guideline Mark-up Tool, or GMT [1, 4, 5]) and Stepper ([2, 3, 6, 7, 8]). Both tools allow the user to define mappings between fragments of two XML documents, providing the necessary support for guideline developers in transforming informal guidelines into more formal representations. Other tools for similar purposes exist, see GEM-Cutter ([9, 10]), or the DeGeL-related tools ([10, 11]), but they focus on other dimensions of guideline development process or do not provide the same flexibility in choosing the target representation model. Our objective is to study the role of refinement and tracking of knowledge objects in producing flexible executable representations of a procedural guideline. By means of a modeling exercise, we investigate the role of these tools, or rather their underlying principles, in producing re-usable knowledge objects which are useful when representing medical processes and performing updates of medical guidelines. We compare the two tools from a knowledge modeler's perspective, according to criteria such as the effectiveness of the model produced, the quality of the representation and the reusability of modeling knowledge across scenarios. We provide a general evaluation of usefulness of the two tools from a reusability perspective and state the main requirements for tools that reuse medical knowledge and support authoring of guidelines.

In the remainder of our paper, we summarize our experiments on applying guideline formalization using Stepper and DELT/A, and provide a preliminary validation of the resulting models. We compare the quality and effectiveness of both transformation tools with respect to producing formal specifications in ASBRU ([14]), an executable plan-oriented language for representing medical protocols. We also explore whether integration of their most useful features is possible.

Both tools were applied on an evidence-based guideline for treatment of breast cancer [12], which provides complex practical situations, suitable to be modeled. We compare the complexity of the transformation process and the quality of the executable model generated by DELTA/A and Stepper, which is given by the number and the complexity of ASBRU plans produced using that tool. Then we summarize the advantages and disadvantages of each tool and the lessons learned from using these tools in guideline formalization.

In DELT/A one typically defines links between parts of two documents which express the same content but in different representation languages. The formalization process is an iterative refinement of various fragments of the text, resulting in an incremental building of an executable model and in mapping increasingly larger narrative fragments to pieces of the model built. As in other mapping and linking approaches, the same transformation step is repeated until no new elements can be formalized.

Stepper uses a multiple-step approach in which the guideline is decomposed into fragments corresponding to a relatively independent set of propositions. These fragments are further split into basic components which can be used to construct an element at the next level of representation of the guideline. Some of these simple transformations can partly be automated and executed independently. Guideline formalization can be viewed as a process consisting of several pipelined refinement steps, performed once for each relevant source fragment, in which only elements produced by the previous refinement step are used.

2 The Step-by-Step Methodology for Guideline Formalization

Stepper ([2,3]) is a tool designed for document formalization based on the idea of multiple-step refinement, in which the focus of each step is to transform the knowledge in the source document into a more formal representation in the destination document. In each subsequent step, the destination document of the previous steps becomes the new source document. The destination element of the last step is typically a formal or executable representation of the original guideline.
A transformation step is described by three components:

1. The schema of the target representation language is a DTD document defining the elements allowed in the target representation
2. The transformation rules file specifies actions (such as, selection, iteration, grouping or splitting) allowed for creating elements in the target representation by using elements in the source file. The Rules file can be created and edited in the Rules editor, which is an integrated module of Stepper.
3. The target representation is an XML file in which elements allowed by the target representation schema are inserted, when the rules for transformation can be applied on the source document received as input.

The guideline formalization supported by Stepper is actually a gradual refinement of the original text into a set of elements that are closer to the formal representation than the original representation. By manually adjusting the types of refinement performed in each step of the formalization and by building a rules file, a

part of the knowledge necessary for performing the guideline formalization task is made explicit. Later, it can be reused for subsequent transformations.

Stepper uses a step-by-step approach for guideline formalization ([2]), transforming the guideline document from the bottom to the top, in a Document-Centric fashion. The goal of the transformation is to preserve the domain-specific and procedural knowledge conveyed by the guideline, while removing the unnecessary bits. This approach differs from the more predominating Model-Centric approach to guideline formalization, in which a compact conceptual model of the guideline is formulated by the domain expert and gradually converted to a fully operational representation.

There are 3 typical steps used in the step-by-step methodology:

1. Split the guideline text into components that contain procedural knowledge and components that contain definitions of concepts; ignore other parts of the guideline;
2. Group together the procedural knowledge elements associated with the same episode, and the definitions they use, making up scenarios;
3. Based on definitions, combine the scenarios produces in previous step into an executable representation of the guideline.

We have customized this process for medical scenarios that have to be translated into ASBRU:

1. Isolate **narrative structures**, i.e. text describing procedural aspects of the guideline
2. Mark up **medical categories**, i.e. medical concepts connected to the procedural aspects
3. Identify **control structures**, i.e. patterns of control that match the selected procedural fragments
4. Sketch a **preliminary ASBRU** representation, i.e. fragments of executable language that can be mapped to procedural fragments using medical categories and control structures
5. Refine it into the **final ASBRU** representation, i.e. the executable form, obtained by refining the preliminary ASBRU elements through reviewing the control structures and the medical definitions.

The essential change compared to the standard three-step approach of Stepper consists of isolating two types of elements in the "Narrative structures": those which correspond to concepts in the medical vocabulary are obtained by semantic annotation of the narrative fragments using medical thesauri; those describing procedural knowledge are identified using a set of linguistic markers, such as "A {following/after} B", "A {consists of} B", "{if} A {then} B". These elements are then mapped to control blocks of the type "execute action A; then execute action B" or "if condition A is true then execute action B". From these structures a skeleton in the target representation language (in our case, ASBRU) is built. The last step accounts for refining the ASBRU skeleton and integrate its components into an executable representation.

3 Document Linking and Mapping

DELT/A ([1]) is a flexible general purpose tool that facilitates linking and mapping between two XML documents, but has been designed to support guideline formalization ([4]). By viewing knowledge refinement as a particular case of mapping between knowledge components, the linking feature provided by DELT/A supports translation of plain text guidelines into more formal representations. The user defines links between the original guideline and the formal representation, and has the possibility to see how each element in the original guideline translates into the formal representation, and which part of the original text a formal element is obtained from. The tool provides a plugin for translating guidelines into ASBRU as the main formal guideline representation language. The tool is flexible enough to support virtually any knowledge representation formalism for which a DTD has been defined. The existing version of the tool supports validation of an XML against its DTD, multiple and overlapping links between elements of two XML files, evaluation of XPath expressions, filtering and highlighting of XML elements.

The guideline formalization process is captured in DELT/A by having several one-to-one mappings between text fragments in the medical guideline and elements of the target model, which in fact define one-to-many (decomposition) or many-to-one (aggregation) relations. The step-wise refinement approach of Stepper can be emulated in DELT/A, by defining intermediate representations and mappings between pairs of these intermediate representations. Visualization support for tracing the transformation of one knowledge object across several refinement levels is simpler than in Stepper (which provides a cross-level view of one element), but this is due to the fact that guideline formalization in DELT/A is seen as a cascade process in which knowledge is preserved and the decisions in modeling are only taken based on the elements in the immediately preceding representation, not based on elements from several preceding representations, as in Stepper.

DELT/A allows the use of macros, written in an XML-based macro language, which provide the same functionality as the transformation rules in Stepper – a standard refinement of one source element through several rewriting steps. A macro is a hierarchically ordered group of XML elements that are conceptually connected and are often used together. It can be seen as a template, which captures the most frequently used transformations for one source element and can be reused during authoring of guidelines.

4 Defining knowledge transformations

We evaluated the suitability of DELT/A and Stepper tools in clinical guideline formalization, by comparing the ease of use and the quality of the ASBRU executable model generated for a few specific scenarios. One additional object of our investigation was the balance between the domain-specific knowledge and the modeling specific knowledge which is required to perform the formalization of a narrative medical text. We have implemented the knowledge transformation process

using the means provided by these tools and studied the gain in effectiveness and quality of representation when this process was repeated for a similar document.

DELT/A and Stepper are both document-centric approaches. Due to the step-wise refinement method used by Stepper, the quality of the target representation (level of detail, presence of complex control structures) depends more heavily on the choice of the intermediate steps taken for refinement. DELT/A can be used to produce quicker a prototype if the mapping between text and elements of the executable language is trivial, because the linking of source and destination elements can be done in one step. However, DELT/A should be regarded as operating at a lower level of conceptualization than Stepper. From this perspective, DELT/A is less semantics-aware than Stepper. Using only one-step for transforming text guidelines into a specification in an executable guideline language (such as ASBRU), as DELT/A does, is difficult. DELT/A is not aware of the different types of control components embedded in one another, while in Stepper this distinction can be made with the right design of refinement steps. However, due to the separation of mapping from semantics, DELT/A gains in flexibility, and there are very few things that Stepper can do which cannot be emulated by using DELT/A macros. Even a multiple step knowledge transformation can be emulated successfully in DELT/A, therefore the two tools can produce comparable results. We want to evaluate which features make them produce better results, which knowledge transformation rules are commonly applied when using them both and whether lessons can be learned about which of these features can be combined when building new applications with similar goals.

Two guidelines written in English were selected for formalization, as they capture recent, clinically relevant, multi-disciplinary, evidence-based material, including recommendations based on expert consensus and based on clinical trials and literature review:

1. the SIGN Scottish [13] guideline for treatment of breast cancer in women.
2. the CBO Dutch ([12]) guideline for treatment of breast cancer.

By analyzing two different guidelines we can assess the differences in the complexity and quality of the transformation with different input text. The parameters that define the quality of the transformation are:

1. the number of anomalies in the guideline knowledge found using the tool;
2. how easy it is to produce an element of the executable model from a source element;
3. the time spent to transform all relevant source elements into executable elements;
4. the quality of the executable model generated (i.e., whether the resulting ASBRU model contains sufficient information to allow simulation of the medical process modeled).

Two narrative chapters were extracted from the SIGN guideline and formalized, first in Stepper, then in DELT/A. Both results could be validated in DELT/A against the DTD of ASBRU. Subsequently, one chapter from the CBO_BC guideline has been formalized, first in DELT/A then in Stepper, using the background knowledge and the project structure collected in the first project (SIGN guideline). This order was chosen to see whether knowledge of the transformation process can be transferred between the two tools and between two different projects of the same

tool. This is motivated by the need to investigate the reusability of formalization knowledge acquired in each tool.

4.1. Knowledge Transformations in Stepper

We defined a guideline refinement scenario consisting of 5 transformation steps: Guideline Document (Source) → Narrative Structures → Medical Categories → Control Structures → ASBRU Elements → Final ASBRU representation. This refinement scheme corresponds to decomposition of linguistic structures that make up the original (plain) text of the guideline, according to three dimensions:

a. narrative categories (e.g., definitions, conclusions, recommendations)
b. detailed conceptual (domain-specific, i.e., medical) categories (medical interventions, drugs, symptoms, diagnostic)
c. control structures: action sequencing, decomposition, synchronization and then reconstruction of the formal representation starting from control structure elements.

Narrative Structures are elements such as: definition, recommendation, conclusion, background fragment (containing explanations, evidence supporting a particular treatment), goal statements (expressing medical intentions), procedural fragments (detailing how the procedures are to be done, in which order, what components they have), recommendations, references to literature or to other parts of the document, medical statements. In the subsequent steps we distinguish several refinement procedures applied to each type of element, and focus on the more "formalizable" elements, such as procedural fragments, conclusions, recommendations.

Medical Categories are elements intended to reveal the knowledge composition of the narrative-control structures. Text fragments that have a well-established structure are further decomposed into categorized terms that belong to medical categories. Extracting medical categories from each narrative structure corresponds to a semantic tagging of the text fragments. Medical representation contains the following medical concepts: med_concept | med_goal | med_factor | lab_parameter | med_action | drug | body_part | disease | time_spec | op_relation | med_reference

The transformation from the narrative structures into control ones can take place directly if this step is skipped, but this semantic tagging potentially reveals knowledge gaps that have to be filled in using background knowledge. For instance, it can be detected that a sentence describing a medical action does not refer to which body part it is to be applied, or which drug it uses.

The next step is to extract the elements that describe different programming-like control structures, using a list of words that describe decomposition, aggregation, ordering of actions, temporal constraints and repetition. The Control Representation contains one of these elements: action-goal | condition-action | action-group | scenario | causal-rel | concept-def | ref-to | action-effect | cycle | iteration | action-time | action-effect.

We selected this transformation step, as each of the programming-like structures has a quite different translation in terms of the formal representation language chosen by us, namely ASBRU.

ASBRU Elements step includes all elements of the formal representation of the guideline in ASBRU. However, the resulting ASBRU set of plans does not include yet optimizations, such as, having a minimal number of plans, unique variable names, disjoint variable and plan names, etc used. The last transformation step, from ASBRU elements to Final ASBRU representation, is a refinement and optimization step.

The Preliminary ASBRU and Final ASBRU representations have the same schema, allowing all ASBRU constructs: local and global variable definitions, plans, effects of plans, intentions and plans and termination and completion conditions for plans.

Rules for transformations necessary for guideline formalization were produced on-the-fly, based on the type of knowledge encountered in the chapters analyzed. After the first chapter of the reference guideline was processed, a few knowledge transformation rules were created.

For instance, for processing of the following text fragment:

Definition: Locoregionally advanced breast cancer is used to describe breast cancer which is unresectable on the basis of the classic unresectability criteria: oedema of the skin (peau d'orange), ulceration, satellite skin nodules, etc.

we added the following transformation rules:

Narrative.rules:

Text → One-to-one (Definition) → Definition

Medical.rules:

Definition → Decomposition (D_Definition2List) →
Seletion([MedConcept;OpRelation;Disease;OpRelation;Iteration(0,5,Med_Symptom)])

Control.rules:

Selection([MedConcept;OpRelation;Disease;OpRelation;BodyPart]) →
Aggregation (A_List2ConceptDef) → Concept-Def

Pre-asbru.rules:

1. Concept-Def → One-to-One (O_ConceptDef2ContextDef) →
   Plan-library\Domain-Defs\Domain\Context-Def
2. Concept-Def → One-to-One (O_ConceptDef2VarDef) →
   Plan-library\Plans\Plan-Group\Plan\Value-Def\Variable-Def

Asbru.rules:

Plan-library\Plans\Plan-Group\Plan\Value-Def\Variable-Def →
One-to-One (O_VarDef2ContextDef) →
Plan-library\Domain-Defs\Domain\Context-Def

Based on the usage patterns observed in the transformation process, we adjusted the elements of the transformation, for instance by partitioning the original "Background" element (class) in the "Narrative" representation into two elements: "Definition" and "Procedural Fragment". In the example above, we created the transformation rules allowing for two options for transforming a concept definition ("ConceptDef" element at "Control" level) into different elements at the "ASBRU" level: a definition of a variable that is going to be used at the level of one plan (Variable-Def), or a definition of a context that will be used by several plans. The role of the definition will likely become clear only after all ASBRU plans have been defined, and the dependencies between plans and the variables have been established. Therefore, this transformation of a variable definition into a context

definition can take place only in the last refinement step ("ASBRU elements" → "Final ASBRU").

By counting how many elements of each type and which rules were used most often in the formalization of the first chapter of the SIGN guideline, we were able to eliminate the less generic and infrequently-used rules and to select the most frequent ones as transformation patterns.

The formalization of the subsequent chapters took much less time, due to the fact that the elements of the intermediate representation and the transformation rules defined in the formalization of the first chapter were re-used. The time spent for formalization using the Stepper tool has been reduced considerably, when enough transformation rules defined in the first phase could be applied in the second phase. By simulating the Stepper transformations in an application that performs semantic tagging of text and applies automatically the transformation steps (for a given order of the refinement steps, and for a given set of elements and transformation rules), we are able to measure the time spent for formalization. On the other hand, the quality of the resulting formal representation in ASBRU is not as high as that of a manual formalization.

For estimating this quality of the resulting ASBRU representations, we used the following criteria:

1. The number of ASBRU plans generated should be as low as possible;
2. Each plan should have the body part and activation conditions defined
3. Each plan should be executed by another plan, or references to its corresponding medical actions must exist in the background narrative description this plan was generated from.

4.2. Knowledge transformations in DELT/A

Two chapters of the SIGN guideline were formalized in DELT/A separately, in two steps. The initial target representation file was initially empty, and new ASBRU elements were added as they were recognized in the source document. The mappings used the reference ASBRU DTD specification and a set of DELT/A macros built for Text-to-ASBRU transformations.

The formalization was incremental in the sense that different parts of the guideline could be transformed independently and later further refined, based on additional knowledge produced in the transformation. Creation of macros took place in parallel with the formalization, so that, after the first chapter was formalized, the most frequently encountered transformations were stored as DELT/A macros. Therefore, the second chapter was formalized using additional background knowledge acquired in the formalization process of the first chapter. As result, the knowledge engineer produced a more finely-grained and complex model than that of the first model built.

Formalization in DELT/A produced better qualitative representations than Stepper, within a time frame comparable with the one spent in Stepper for the same input. For small and finely structured knowledge objects, the possibility to track directly a formal element to its context in the source document was essential in obtaining a detailed formal representation. This feature was more prominent in

DELT/A, where there was only one formalization step to perform, while the cross-level visualization feature in Stepper helped towards this goal, but involved additional actions.

4.3. Comparing the effectiveness of guideline formalization tools

We formalized the same guideline text extracted from two guidelines in each of the tools, then we cross-validated the results obtained with each tool, with the help of the other tool.

We noticed that in DELT/A more ASBRU plans can be identified than in Stepper, as the transformation process is more direct and the original information is not lost in transformation. This is why we used an independent mark-up of the guideline text to verify that all the relevant parts of text have been processed with both tools. The evaluation has been done based on the several criteria, discussed below.

4.3.1. Ease of use. We evaluated how easy is to set up a refinement scenario and to perform the guideline formalization without previous training. With Stepper it is conceptually clearer to set up a refinement scenario because it is aware of knowledge refinement and splits the guideline transformation process into logical steps. It makes guideline formalization easy without previous training because knowledge is processed at different levels of abstraction or even by different categories of people (medical experts, knowledge engineers, guideline developers). In DELT/A defining the refinement scenario takes more time, but using the macros in formalization is relatively easier than applying the transformation rules in Stepper.

4.3.2. Richness of transformation. The granularity of the elements used during guideline knowledge refinement and the types of transformation rules that can be described determine how expressive the transformation is. In Stepper, the granularity of the elements resulting from transformation is finer than the one used in DELT/A, therefore potentially the formalization in Stepper can capture more formalization knowledge than DELT/A. In Stepper more complex transformation rules can be defined than in DELT/A, such as selection of elements, aggregation of several elements of different types, iteration, and repetition of a particular processing.

4.3.3. Quality of the resulting executable model. It depends on the granularity and complexity of the executable model obtained through transformation. With Stepper, the transformation is a composition of several aspect-oriented transformations at different conceptual levels. This compositionality ensures a more refined result, compared to the more direct knowledge transformation (in one step) in DELT/A. However, DELT/A copes better with changing knowledge, as it does not depend on a long knowledge refinement chain and the changes can be applied faster.

4.3.4. Visualization support. To evaluate how intuitive and easy to use is the visualization support in the two tools, we looked at how easy is to navigate, whether focus on particular parts of guideline is possible while others are hidden, whether several regions of the guideline can be analyzed concurrently, compared, etc. Stepper provides a very convenient cross-level view of an element, which indicates the transformation tree that produced that element, including the order of transformations that have been applied. Different perspectives are given to the

modeler, to focus on particular components of the model which is built. In DELT/A, the graphical representation is replaced by a tree-like textual view in which particular language constructs and XPath expressions can be input and matching elements highlighted. However, the focus is on document structure, and does not look from modeller's perspective. The order of modeling steps is not captured. However, from an extensibility perspective, DELT/A is more powerful, as it comes with a set of plugins that can be extended / rewritten to generate graphical representations, though this is not directly supported by the tool itself.

4.3.5. Support for explanation and reasoning. Both tools provide an explicit visualization support for explaining the guideline transformations. Traceability of the steps performed (step history) and connecting the guideline parts with a particular semantics are important in having a better support for understanding the modeling. The Stepper tool provides a feature called "Cross level view" that gives a graphical representation of the transformations done for obtaining a particular element, while DELT/A keeps a counter for recording the order of transformations. However, DELT/A provides a set of views, filters, search and XSLT transformation features which make it potentially richer from an explanation and reasoning perspective.

4.3.6. Facilitating re-usability and modularity of knowledge. An essential feature common to the two tools is the support offered for re-using the transformation steps capturing guideline modeling decisions in other situations, with a minimum of change. In Stepper, the fine granularity of the knowledge captured by the transformation rules allows several complex transformations to be realized by composing several basic transformations. These transformation rules used in each step of the formalization are defined and customized by the knowledge engineer and kept separately in a rules file, in order to be re-used in other translations [4].

5 Conclusions

We compared the implementations of two knowledge transformation principles for guideline formalization, embodied by two of the very few tools that can be used in practice for improving the quality and effectiveness of guideline formalization. The two applications discussed, DELT/A and Stepper, were designed for different purposes: Stepper is a multi-step, customizable document transformation tool, designed for step-by-step knowledge refinement, while DELT/A is a multi-purpose, macro-based, document mapping tool. They have a lot of common, as well as complementary, features, compatible input and output formats (XHTML, XML, DTD) and one common method – cross-document mapping.

The results obtained by formalizing the guidelines with these tools represent valid representations in the target ASBRU language, but still require additional adjustments in order to be executable by an ASBRU interpreter. This indicates that either the text formalized did not contain sufficient procedural knowledge, or the information lost during transformation affected the quality of the ASBRU representation generated. However, the positive impact of the structured guideline development process that these tools implement, on the effectiveness and the quality of the guideline representations generated cannot be dismissed. In the future,

separation of domain knowledge from modeling knowledge and maintenance of guideline modeling rules as re-usable components will lead to libraries of pre-compiled transformations that can be imported, tested and applied to specific categories of documents.

The features discussed here represent a few steps in this direction. The refinement process proposed by Stepper and the use of DELT/A macros reduce significantly the time spent in formalization of a guideline for which a guideline formalization process has been configured. Also, the possibility to track a formal element back to its original context in the source document is decisive for a quick understanding of modeling steps and consequently for a better quality of the representation.

As result of our comparison, several requirements can be listed for a guideline formalization tool that would combine their most useful features:

1. execution of multiple, logically independent, **knowledge refinement steps**, which can be traversed both forward (source-to-target) and backward (target-to-source). Stepper supports **visual tracing of the refinement steps**, while in DELT/A a refinement scenario consists of multiple transformations, each involving a pair of documents, and can be reviewed in a pair-wise fashion;
2. **knowledge visualization support** for analyzing the origin, semantic category, importance and precision of data chunks; currently, the history file and the cross-level view in Stepper provide such support, but it can be included in a more natural way as plugin for DELT/A;
3. the **use of macros** (present in DELT/A, absent in Stepper), i.e. the most frequently used transformation rules by which elements in the source representation are transformed into target representation elements;
4. possibility to **customize and iterate the refinement process**, by allowing destination elements of the current step to be included among the source elements of this step; currently, both Stepper and DELT/A support this process
5. **modeling decision support**: user assistance for viewing, editing, summarizing and visualizing available transformations, **support for validation** (checking consistency of an XML file versus its DTD, filtering and highlighting of missing links, etc); DELT/A provides a comprehensive support for this, in Stepper tool new elements can be inserted at the right place in an XML file only in accordance to the DTD schema of the target representation language;
6. **support for semantic tagging and ontologies** in the refinement process, including script support for reporting, refactoring and transformation of parts of the edited documents; Stepper promotes XSL for automating the knowledge transformation, but so far DELT/A has more advanced features concerning the interactive transformation and refactoring of the medical knowledge fragments.

The design principles proposed by the Stepper methodology for guideline formalization provide useful insights for structured guideline development process. For implementation purposes, DELT/A tool is more effective and can be extended more straightforwardly. However, both tools provide many useful and complementary features, which can be reconciled by an application supporting guideline formalization through an incremental refinement of knowledge.

References

1. Votruba, P - Structured Knowledge Acquisition for ASBRU. TR, Vienna University of Technology
2. Ruzicka, M.; Svatek V. - Step-by-Step Mark-Up of Medical Guideline Documents, International Journal of Medical Informatics, vol 70, number 2-3, July 2003, pp 329-335
3. Ruzicka, M.; Svatek V. - An interactive approach to rule-based transformation of XML documents, Proc. DATAKON2003, Brno, The Czech Republic, October 2003, pp. 1-10
4. Votruba, P.; Miksch, S.; Kosara. R.: Tracing the Formalization Steps of Textual Guidelines, in Kaiser, K.; Miksch, S. and Tu, S. (eds.): Computer-based Support for Clinical Guidelines and Protocols. Proc. Symposium on Computerized Guidelines and Protocols (CGP 2004), volume 101, Studies in Health Technology and Informatics. IOS Press, 172-176, 2004.
5. Votruba, P.; Miksch, S.; Kosara, R.: Linking Clinical Guidelines with Formal Representations, in Dojat, M.; Keravnou, E.; Barahona, P. (eds.): Proceedings of the 9th Conference on Artificial Intelligence in Medicine in Europe (AIME 2003), Springer, Berlin, pp. 152-157, 2003.
6. Svátek, V.; Kroupa, T.; Ruzicka, M. - Guide-X – a Step-by-Step, Markup-Based Approach to Guideline Formalisation. Leipzig 13.11.2000–14.11.2000. In: HELLER, B; LOFFLER, M; MUSEN, M; STEFANELLI, M. (eds). Computer-Based Support for Clinical Guidelines and Protocols. Proceedings of EWGLP 2000. Amsterdam : IOS Press, 2001, s. 97–114. ISBN 1-58603-193-7.
7. Ruzicka, M. - XML Knowledge Block Transformation (XKBT). Online at http://euromise.vse.cz/stepper/xkbt
8. Ruzicka M.; Svatek V. Mark–up based analysis of narrative guidelines with the Stepper tool. Proc. Symposium on Computerized Guidelines and Protocols (CGP-04), Praha 2004. IOS Press.
9. A. Agrawal and R. N. Shiffman, Using GEM-encoded guidelines to generate Medical Logic Modules. AMIA 2001.
10. Y. Shahar - A Hybrid Framework for Representation and Use of Clinical Guidelines, in Proc. AMIA 2002, San Antonio, Texas 2002.
11. Shahar, Y; Young, O; Shalom, E; Mayaffit, A.; Moskovitch, R; Hessing, A; Galperin, M - DeGeL: A Hybrid, Multiple-Ontology Framework for Specification and Retrieval of Clinical Guidelines. Proc. of the 9th Conf. on AI in Medicine—Europe (AIME) '03, Cyprus, Oct. 2003, Springer-Verlag Heidelberg, pp. 122 - 131.
12. CBO. Guideline for the Treatment of Breast Carcinoma. CBO, 2002. PMID: 12474555.
13. Scottish Intercollegiate Guidelines Network. Breast Cancer in Women, SIGN#29. SIGN, 1998.
14. Johnson, P.; Miksch, S.; Shahar, Y. - Asbru: A task-specific, intention-based, and time-oriented language for representing skeletal plans. In Pierret-Golbreich C.; Filby I.; Wijngaards N.; Motta, E.; Harmelen, F.v. (eds), Proceedings of the 7th Workshop on Knowledge Engineering: Methods and Languages (KEML-97), 1997.
15. Balser, M.; Coltell, O.; van Croonenborg, J.; Duelli, C.; van Harmelen, F.; Jovell, A.; Lucas, P.; Marcos, M.; Miksch, S.; Reif, W.; Rosenbrand, K.; Seyfang, A.; ten Teije, A. , Proc. Symposium on Computerized Guidelines and Protocols (CGP-04), Prague, Apr. 2004. URL: www.protocure.org.