Mark–up based analysis of narrative guidelines with the *Stepper* tool

Marek RŮŽIČKA, Vojtěch SVÁTEK
European Centre for Medical Informatics, Statistics and Epidemiology - Cardio,
University of Economics, Prague, W.Churchill Sq. 4, 130 67 Praha 3, Czech Republic
E–mail: {ruzicka, svatek} @vse.cz

Abstract. The *Stepper* tool was developed to assist a knowledge engineer in developing a computable version of narrative guidelines. The system is document–centric: it formalises the initial text in multiple user–definable steps corresponding to interactive XML transformations. In this paper, we report on experience obtained by applying the tool on a narrative guideline document addressing unstable angina pectoris. Possible role of the tool and associated methodology in developing a guideline–based application is also discussed.

1 Introduction

Since long time, the community of computerised guideline research has been aware of the conceptual gap between textual version of guidelines and its operational counterpart. This phenomenon, important even for documents containing flowcharts and tables, becomes critical for exclusively textual, narrative documents, which typically stand at the beginning of the guideline authoring process. Recently, following the rise of XML technology, *mark–up editors* appeared in the guideline domain. Two different ways of their use are tied to two concepts of guideline formalisation: model–centric and document–centric. *Model–centric* tools assume a compact model of the guideline (based e.g. on branching logic or skeletal planning) to be built first, and only link selected textual fragments to it for the sake of documentation: the process is thus top–down. Conversely, *document–centric* tools take the original text as starting point: they build the guideline model bottom–up, through filling marked–up content into knowledge containers (of pre–defined categories). An advantage of document–centric approaches is better coverage of information that does not fit well into the compact model (esp. broader context of clinical care), and lower risk of leaving an important piece of information unspotted in the original text. Their main disadvantage is however difficulty of bridging the above–mentioned semantic gap: models built bottom–up typically stay on the half–way to fully operational application, since the textual content, though delimited with tags, still has to be interpreted by a human.

In this paper we present a document–centric mark–up tool called *Stepper*, which addresses the mentioned problem by decomposing the guideline formalisation process into multiple user–definable steps corresponding to interactive XML transformations. Section 2 briefly describes the functionality of *Stepper*. Section 3 demonstrates its use in connection with a typical narrative guideline document: the Czech guidelines for unstable angina pectoris. Section 4 compares *Stepper* with similar mark–up editors. Section 5 discusses its possible role in a broader context of guideline–based application development. Finally, section 6 outlines directions for future work.

2 Overview of *Stepper*

The system consists of multiple interconnected user environments:

- for free–text mark–up (delimitation of initial ‘knowledge’ blocks)
- for interactive step–by–step transformation of XML structure and content
- for easy navigation along links (between source and target structures) across all transformation levels.

Stepper contains an embedded XSLT [2] processor, which carries out the *non–interactive* part of the transformation, while the *mark–up* and *interactive* transformation are carried out by means of rules expressed in a new language called XKBT [6]. Mutually corresponding XML structures at different levels are linked via XLink [3] references, including the source document, for which XHTML format is actually used.

The most conspicuous feature of *Stepper* is the capability to explicitly decompose different phases of document formalisation. In [10], we suggested as many as five transformation steps needed to obtain a fully operational knowledge base from a free–text guideline document; part of this methodology has already been incorporated into a sequence of Document Type Definitions (DTDs), which can be interpreted by *Stepper*. The view of formalisation as stepwise process naturally leads to breadth–first traversal of logical document structure: the knowledge engineer who analyses the document is encouraged to first observe the global structure of discourse, and only then concentrate on details of individual statements. Explicit decomposition, supported by semi–automated processes, should make the individual steps on the way from document to operational knowledge base small enough to become manageable.

3 Marking up in *Stepper*: experience with unstable angina guidelines

In the phase of system development, we tested *Stepper* on fragments of WHO/ISH hypertension guidelines – a document with many tables and flowcharts. For a more complex evaluation, we however chose a more narrative document: guidelines for unstable angina recently developed by the Czech Society for Cardiology. The generic guideline model we use in our experiments was described in more detail in [9]. Each instance of the model is actually a collection of components belonging to four categories: *procedural* component (roughly identified with the notion of scenario), *causality*, *goal statement* and *concept definition*. Implementation of the model in *Stepper* respects its stepwise nature; it consists of multiple interconnected models (in the form of DTDs) reflecting the evolution of components during transformation. In the first phases, transformations mostly have the character of refinement, since elements are gradually provided with tree structures of sub–elements; later, some of them (goals, scenarios) can also be e.g. aggregated.

In the mark–up environment of *Stepper*, we went through the whole document (about 30 kB of text), and assigned selected fragments of text to the four categories of components mentioned above. As we see at Fig. 1 the category is chosen by pressing one of buttons in the upper–right pane, generated in runtime from the premises of applicable XKBT rules. A corresponding node then appears in the tree (lower–right pane). Conversely, selecting a node of the tree highlights the corresponding text in the document pane. Extracted text is displayed (and editable) in the lower–right pane, interchangeably with attribute form (derived from the DTD). The whole process took about one hour of work of a knowledge engineer with limited medical background; it yielded a (flat) XML structure with 64 elements. Of them, 15

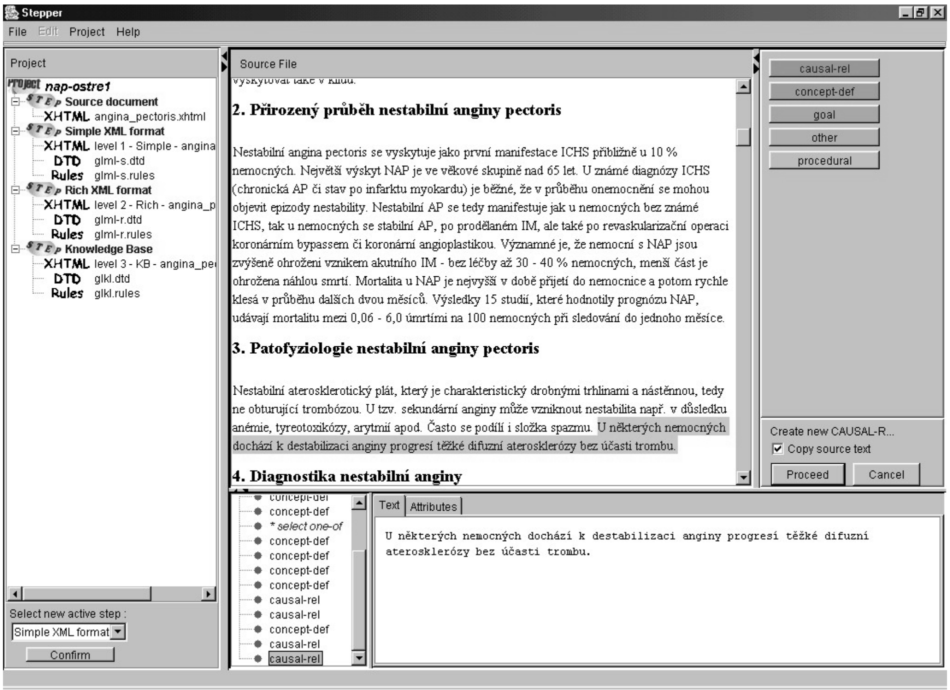

Figure 1: Extraction of basic components in step 1

corresponded to concept definitions, 12 to causalities, 35 to procedural components and 2 to goals. About 25% of the original text was discarded as unusable.

We also, tentatively, transformed some components of the document to the next level. In the 'XML–to–XML' mode, the upper part of the screen shows the source level and the lower part the target level of transformation; each of them again consists of an XML tree and a pane for editing the text content and attribute values. Fig. 2 shows the process of transforming a free–text 'procedural' element into a structure of 'scenario' consisting of a condition and a recommendation part, the former corresponding to a potentially complex expression over patient states and/or history; the currently edited 'predicate' element ("pain duration greater than 20 minutes") defines an elementary patient state. The whole structure was generated by an XKBT rule fired by pressing the 'Scenario' button; the button only appeared upon selecting a 'procedural' element in the upper tree.

We plan to push the whole document to the next level in the near future. However, efficient support to finer–grained formalisation will only be provided when an envisaged *clinical terminology* plug–in to *Stepper* (now in the stage of conceptual design) comes into being.

4 Related work

The prominent approach in 'document–centric' guideline formalisation is *GEM* [8]. The process of GEM encoding consists in populating a rich and structured collection of elements with content extracted from the text. This approach is reflected in the associated tool called *GEM–Cutter*[1], which supports easy navigation in the XML tree, and displays the extracted text directly within its nodes. Although their user interfaces look similar at the first sight, *Stepper* differs from *GEM–Cutter* in several aspects:

- In addition to initial text mark–up, it provides separate interfaces for 'XML–to–XML' steps. While *GEM–Cutter* assumes a one–step (though iterative, by deliberation of the

[1] Available from `http://ycmi.med.yale.edu/gem`.

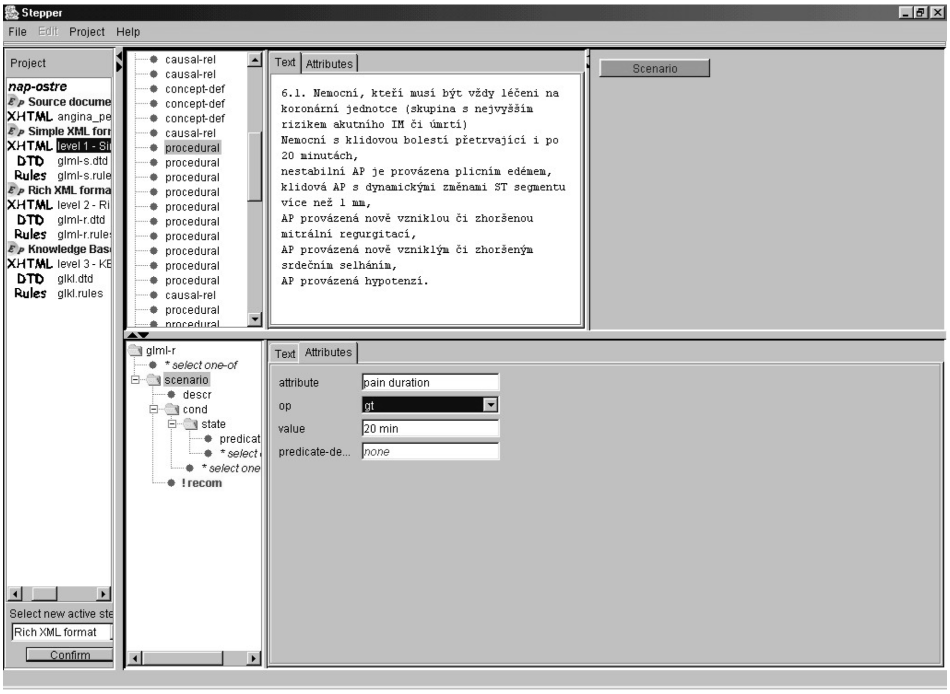

Figure 2: Refinement of a procedural scenario in step 2

user) transition from text to fully populated model, *Stepper* enables to distinguish the steps explicitly, and partially automate them by means of transformation rule sets.

- *GEM–Cutter* was developed specifically for GEM, which is an extremely rich and complex model; hence, the tool must enable navigation over a large part of the resulting XML tree. *Stepper* was developed in connection with a model consisting of small, self–contained components, which are mostly created and transformed sequentially. This distinction influenced the ergonomy of both tools.
- *Stepper* also seems to make more extensive use of XML attributes, and provides linkage from elements to original text (rather than one–way text extraction).

For comparison, we also experimented with marking up our unstable angina guidelines using *GEM-Cutter* and thus GEM as underlying model. In this way, we could cover (mostly non–operational) information present in the document, which was beyond the reach of our simple model (e.g. the identity of guideline developer). Such information is however not the target of formalisation in our setting.

Recently, a method was designed for transformation of GEM–encoded guidelines to Arden syntax [1], using an XSLT [2] style sheet. Analogously, in our earlier experiments with hypertension guidelines [10], we developed a computable representation (collection of Java classes) in *Stepper*, via XSLT–based export from the last in a row of evolving XML–based models. Our experiment however only concerned a tiny fragment of the original guideline.

Other examples of guideline mark–up tools are *GMT* [4] and *DeGeL* [7]. Both are associated with the Asbru model/language, offer some level of model–to–text linkage with search facility, but only support a single step of formalisation.

5 Discussion

Although the stepwise approach to guideline formalisation will hopefully eliminate some shortcomings of the document–centric paradigm, we do not believe a complete decision–support application can be built in such a bottom–up manner. Namely, the structure of text intended for a human reader can hardly be tweaked to the form of optimal computable model merely via (albeit sophisticated) XML transformations. *Stepper*–like mark–up technology could however facilitate a 'blueprint' run of computable model development. A throw–away computable model developed with minimal involvement of clinical expert could be applied on patient data, and the resulting discrepancies would indicate missing pieces of background knowledge. The same mark–up tool could then assist in adding pieces of knowledge (picked up from the text) to a 'live' model skeleton built top–down e.g. in GLIF [5].

6 Conclusions and future work

In this paper we described the functionality of the *Stepper* system, and its use for mark–up of narrative guideline documents. Its principle of interactive step–by–step XML transformation is expected to enable transition from narrative guideline document to (a simple form of) operational application with limited human effort. The most urgent task connected with *Stepper* is to develop the *clinical terminology plug–in*, which would ease the task of the knowledge engineer and semantically align the target knowledge base with patient data. We would also like to systematically examine the ways knowledge components represented in XML could be *converted* to operational code directly applicable on patient data.

The research has been partially supported by the project LN00B107 of the Ministry of Education of the Czech Republic.

References

[1] A. Agrawal and R. N. Shiffman, Using GEM-encoded guidelines to generate Medical Logic Modules. AMIA 2001.

[2] J. Clark, XSL Transformations (XSLT) Version 1.0, W3C, 1999. http://www.w3.org/TR/xslt.

[3] S. DeRose, E. Maler, D. Orchard, and B. Trafford, XML Linking Language (XLink), W3C Working Draft, W3C, 1999, `http://www.w3.org/TR/xlink`.

[4] R. Kosara, S. Miksch, A. Seyfang, and P. Votruba, Tools for Acquiring Clinical Guidelines in Asbru. In: 6th World Conf. on Integrate Design and Process Technology (IDPT'02).

[5] M. Peleg, A. A. Boxwala, O. Ogunyemi, Q. Zeng, S. W. Tu, E. Bernstam, L. OhnoMachado, E. H. Shortliffe, and R. A. Greenes: GLIF3: The Evolution of a Guideline Representation Format. AMIA 2000 Annual Symposium, Los Angeles, CA, 645–649.

[6] M. Růžička, and V. Svátek, An interactive approach to rule–based transformation of XML documents. In: DATAKON 2003, Annual database conference, Brno, The Masaryk University 2003, 277–288.

[7] Y. Shahar, A Hybrid Framework for Representation and Use of Clinical Guidelines, in Proc. AMIA 2002, San Antonio, Texas 2002.

[8] R. N. Shiffman, B. T. Karras, A. Agrawal, R. Chen, L. Marenco and S. Nath: GEM: A proposal for a more comprehensive guideline document model using XML. JAMIA 2000; 7(5):488–498.

[9] V. Svátek, T. Kroupa, and M. Růžička, Guide-X – a Step–by–step, Markup–Based Approach to Guideline Formalisation, in (B. Heller , M. Loeffler, M. Musen, M. Stefanelli, eds.:) Computer-Based Support for Clinical Guidelines and Protocols, IOS Press, Amsterdam, 2001, 97–114.

[10] V. Svátek and M. Růžička, Step–by–Step Mark–Up of Medical Guideline Documents. International Journal of Medical Informatics, Vol. 70, No. 2-3, July 2003, 329–335.