Supporting Knowledge Modelling by Multi-modal Learning: Defining the Requirements

Katharina Kaiser and Andreas Seyfang

Institute of Software Technology & Interactive Systems
Vienna University of Technology, Vienna, Austria
`{kaiser, seyfang}@ifs.tuwien.ac.at`
`http://ieg.ifs.tuwien.ac.at`

Abstract. A persistent problem faced in modelling medical knowledge is the required combination of skills from the fields of computer science and medicine. The expertise of experts both in medicine and in computer science has many dimensions which makes competence deficits hard to target. Our approach is to provide the required training for the modelling job integrated into the editing environment, providing training on the job and mutual support by modellers with complementary expertise.

In this work, we analyse the requirements of an editing environment that blends various forms of learning with the editing process itself. In particular, we need to combine (1) static online information of various granularity, (2) support for asynchronous interaction with advisors, (3) tracing of user actions and pattern matching of similar use cases in the past to support the learners memory, and (4) automating parts of the modelling process to produce suggestions automatically where possible.

We are currently integrating the above into a system which is expected to provide a productive environment for experienced users and a smooth and efficient learning experience in those subdomains of knowledge where the user is less seasoned.

1 Introduction

Clinical guidelines and protocols are important means to improve the quality of care [1]. To make their application more efficient, they are translated to computer-interpretable models (see [2] for an overview and comparison). This task requires expertise in medicine and in knowledge engineering.

In our knowledge modelling work, we identified three different user groups: Expert knowledge modellers, expert physicians, and middlemen. The first two groups are expensive and hard to recruit. Middlemen can be students of medicine with an interest in formal aspects of knowledge, epidemiologists, and students of computer science with an interest in medicine.

The first group, expert knowledge modellers, are routine users of complex tools such as Eclipse[1] which require significant training. These tools feature dense presentation of

---

[1] http://www.eclipse.org, last accessed March 24, 2011

information which are flexible to use with an enormous amount of features, but require exhausting training and are tailored for high performance users.

The second group, physicians, do not have any modelling knowledge. To communicate models to them, intuitive visualisations are required. Entering knowledge using such an interface can be a longwinded process and may result in a coarse and incomplete output that requires further modelling by a knowledge engineer.

These first two groups are rather theoretical assumptions whereby we come to the third group: The third group, the middlemen, provide the interesting challenge discussed in this paper. Their previous knowledge varies widely, and the depth of this knowledge with relation to the requirements of the modelling task at hand may be uneven. They need to be productive, i.e., performing learnt steps in a direct way. At the same time, they need fallback options for those cases, where their knowledge is not sufficient. These options must be provided on various levels of abstraction and granularity to ensure that the level of support needed for the particular knowledge gap at hand is met.

Since very few persons fully comply with our slightly exaggerated description given for the first two groups, we hope to support parts of these groups, too. Even expert modellers will not know all the details of a representation and providing the suitable piece of knowledge for them where it may be needed will increase their productivity. Likewise, persons without knowledge modelling experience will develop an interest in the basic concepts behind the models presented to them.

2 Background and Related Work

Guideline modelling is in many ways similar to programming. In software engineering (SE) numerous tools and frameworks are provided that support the SE task and often also collaborative and distributed work. Novice programmers come up with mental or cognitive obstacles and misconceptions about computing that make it difficult to understand the functioning of programs or the construction of algorithms [3]. Misconceptions can be attributed to the fact that programmers possibly interpret computer programming as communication among humans [4].

This has a striking parallel in guideline modelling where the freetext version in fact is aimed at human readers interpreting it using their background knowledge while the formal model of the guideline is interpreted by software, using only the knowledge explicitly defined. Kramer [5] considers the ability to abstract as the key to programming. He focusses on two aspects of abstraction: leaving out irrelevant properties, and extracting common properties from a set of instances. This is akin to the situation of a learning knowledge modeller, presented with the challenge to find his own insight into which language element to use under which circumstances. Environments have been developed to support the learning of programming and construction of algorithms (e.g., [6,7]) that feature certain features dictated by their indented didactic use.

For formal and semiformal representations of clinical guidelines and protocols, the modelling problem is generally solved by establishing mixed teams of domain experts and knowledge engineers. Problems communicating the formal aspects and the fundamental logical concepts to persons without IT background are reported, but training approaches are mostly traditional combinations of training lessons and introductory documents. E.g., Shalom [8,9] describes a “Markup-Kit” used to train physicians for the modelling task. It contains an introduction to the modelling tool (URUZ, part of the Degel framework [10]) and to the representation (hybrid Asbru).

Models from different authors are known to be different with knowledge modelling experience and professional background (medical or computer science) being important factors [11]. However, the issue is blurred by the fact that mapping guidelines in natural language which can be ambiguous to concise concepts in a formal guideline, taken from a medical terminology, and adapting these concepts to the situation at the point of care, is a complex process, in which the skills of the modeller are only one of several factors [12,13].

Several means that are based on methods, media, and theories of learning try to overcome problems programmers and modellers, respectively are faced with.

Modes of learning. The classical distinction separates *asynchronous* and *synchronous* learning. The first combines self-study with asynchronous interactions to promote learning, such as email, electronic mailing lists, threaded conferencing systems, online discussion boards, wikis, and blogs. The second means face-to-face meetings.

Asynchronous learning is better suited for bigger audiences, where it can provide cost reduction over synchronous learning. The latter is the better choice for small and heterogenous groups of learners.

Communication media. *Online media* serve the communication between advising experts or teachers and the users or learners, as well as for the communication between the users. Given the varying size of user communities, and the varying availability of experts, it is important to offer a broad range from direct communication with advisors to self-organised communication between the learning. An important means of improving efficiency is a powerful search facility which finds answers given before.

Offline media comprise context-sensitive help in the editor, tutorials and reference manuals. The latter two can be provided both in electronic and paper versions for cognitive reasons. However, the user will always want to have the software at hand to try out the read pieces of knowledge. This not only supports the memory, it also helps retrieve the learnt items if they are forgotten, thanks to the logging mechanism of such system.

Personalised learning experience. Learning theories can be categorised in *behaviourism*, *cognitivism*, and *constructivism*. In behaviourism learning is the acquisition of new behaviour through conditioning (e.g., by applying direct instructions or drill & practice). Cognitive theories look beyond behaviour to explain brain-based learning and consider how human memory works to promote learning. They can be supported by tutorial and adaptive systems. In constructivism the learner actively constructs or builds new ideas or concepts based upon current and past knowledge or experience, e.g., by simulations or games.

3 Analysing Modellers' Problems

As explained in the previous sections, perfect modelling experts as well as medical experts without any modelling background are rare. In practice, almost everyone is on a learning path from very limited understanding to near complete knowledge of a certain representation and methodology.

Based on our experience in teaching medical informatics students in courses and modellers in various research projects to model guidelines and protocols, we summarise the major problems as following:

Principles of guideline modelling. There are many common misunderstandings or misconceptions concerning the principles of programming and the construction of algorithms among guideline modellers. They resemble problems reported for students learning programming [3,4]. Examples for questions arising are: Which parts of the text must I map in the representation formalism? How can I translate text to rules? When do I have to model parameters or variables? What is a decomposition and when do I need it? Which information is necessary to have a guideline executed automatically?

Concepts of guideline formalisms and their application. Apart from fundamental principles of modelling and programming respectively, gaps in the knowledge about concepts of a guideline representation formalism and their application are obstacles for modellers. Questions arising are for instance: Which concepts exist? There are too many language concepts to choose from: which is the appropriate one?

Many languages, for guideline modelling just as much as for programming, contain many rarely used features. Some of the learners consider it their fault not to find proper applications for such language elements.

Representation format. Textual formats of most representation formalisms are not appropriate for communicating the guideline logic to humans [14]. For instance, non-IT persons (e.g., physicians) have difficulties understanding the concepts of XML (i.e., elements, child elements, attributes). Decoding an XML tree and reading, editing, or even navigating a plan hierarchy is very difficult.

Modelling tool. Learning to work with an application demands effort, which many people are not able to spend. Problems modellers are faced with are: How does the application structure the workflow? Where do I find a certain tool/feature/option/... in the application?

4 Solution

In order to overcome the problems described in the previous section we propose several means that are based on different methods, media, and theories of learning.

Since each partial solution has its specific cost and limitations, it is crucial to combine them, and to leave the decision how much of each offering is actually deployed to the user.

Based on these foundations we take the approaches described in the following to overcome the observed problems and to support guideline modellers. There are two parts of a solution. One is to provide an appropriate guideline modelling tool that supports intuitive and easy handling. The other is to offer training and help, embedded into the editing process, as well as in preparation to the modelling task. In this paper, we focus on the second part. We draw the inspiration from the field of SE and various forms of eLearning as used in Blended Learning and Learning on Demand.

Support to user groups is provided in a four-fold manner. Thereby, different kinds of information are provided in different ways to different users. See Figure 1 for an overview.

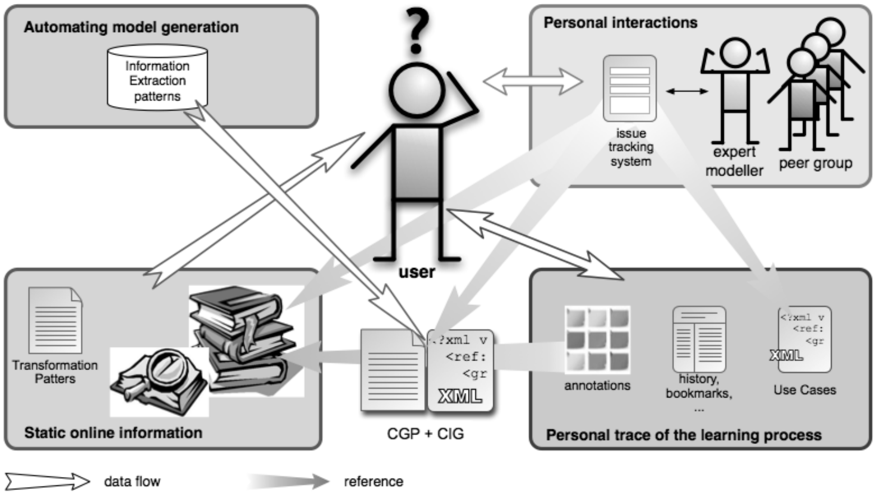

Fig. 1. System components and their interactions. The user is supported by (1) static online information, (2) personal trace of his/her learning process (e.g., via bookmarks or annotations), (3) automating parts of the model generation, and (4) personal interactions with experts and peers.

4.1 Static online information

The simplest part of the system is the provision of static information from three different perspectives. Looking bottom-up, we provide context-sensitive description at the place and time of data entry. Top-down, we introduce the language element and the fundamental concepts behind them, as well as the principal modelling workflow and the associated features of the modelling tool. Complementing both views, we offer a transformation pattern-based approach to the modelling task.

Context-sensitive description. This is useful on two levels, representation-specific and tool-specific. For each element of the representation as well as for each of its knowledge slots, a short description is supplied. More detailed information should then be given on demand showing either the most elaborate versions of the summarised explanation or by giving an example of the usage.

Systematic concept introduction. In addition to providing a manual explaining all concepts in detail, we must be able to provide integrated information for the user on several different levels, without slowing down the working routine of those who already have the required knowledge.

This information is grouped hierarchically according to the concepts of the representation. In addition, it has to be grouped in two layers: (1) abstract concepts described in a language-independent way (e.g., “What is decomposition?”), and (2) language-specific implementation of these concepts (e.g., “How is a decomposition applied in Asbru?”).

In a similar manner, the foundations of the workflow using the editing tool have to be described in a structured way.

Transformation patterns. At least some parts of the modelling task can be seen as mere transformations of patterns in representations. Patterns similar to those defined by the Workflow Pattern Initiative[2] can help identifying a specific pattern and transforming it to the target model.

Focusing on the detection of such patterns – beyond citing them as examples for single syntax elements – means to take an input-driven approach that complements the output-driven approach taken by the two offerings described above.

These patterns will be defined in a way easy to match for humans, but less suitable for an automatic model generation using Information Extraction (IE) methods. Some of them might be formulated concise enough to be detected automatically. However, the important point is to explain the idea behind them to the human user to enable him/her to apply the patterns even when IE fails to detect it, which can easily happen due to gaps in the ontology used.

4.2 Personal trace of the learning process

Using the means described in the previous subsection, the user builds up a knowledge base gradually. Remembering the learnt things is one of the main burdens in this process. This can be greatly reduced by storing all user interactions and providing multiple views on them.

Page visits, bookmarks and annotations. Reallocating information that had been found is important for users. This can be supported by a history and complemented by bookmarks. In addition, it is necessary to permit the user to make annotations at each of the elements. These can be collected by the author of the original documentation to find potential to improve it.

History of use cases. For any syntax element, the user can retrieve a list of cases where he/she used this element in the past. In addition, a list of such use cases by other team members can be generated. Both options together provide personalised usage

---

[2] http://www.workflowpatterns.com, last accessed June 4th, 2011

examples which are highly likely to match the next potential use case. The second option is also important to align the modelling styles of different modellers to each other.

4.3 Automating Parts of the Modelling Process

In order to reduce the modelling effort of the human modeller Information Extraction (IE) methods can be used to automate parts of the modelling process.

We use IE for two different purposes. First, it is used to replace the modelling effort of the human modeller as far as possible by generating translations from the source text or model to the target representation, wherever this can be performed automatically. Clearly, the result must be validated by a human modeller, as always in IE.

Second, we use IE to detect patterns in the input which cannot be analysed complete enough to suggest a translation, but where there are hints at the application of certain patterns. While the transformation-based approach of knowledge communication to the user described in the previous subsection takes a rather top-down approach focusing on human cognition, suggesting possibly interesting elements by means of IE takes a bottom-up approach focusing on technical possibilities of automatic detection.

4.4 Personal interactions

Expert assistance. There are always situations where the above aids are not sufficient and a human expert must be contacted. In the field of modelling, there are cases where more than one solution is acceptable and the modelling team needs to gradually come to a conclusion which solution to adopt. To keep communication efficient in these cases, we want to provide means that resembles features from an issue tracking system.

Peer assistance. Not all the questions which a user cannot answer by reading the available information need a reply from a designated expert. In practice, the knowledge of medium advanced learners will complement each other. They are more likely to be available, and discussing the learnt subject helps all participants to reinforce it and avoid misunderstandings.

5 Discussion

5.1 Matching Modellers' Problems with the Solution

The solutions described in the previous chapter help facing the various learning challenges described in Section 3 following a complex pattern shown in Figure 2.

The strength of *context-sensitive descriptions* lies in showing the details for both representation and modelling tool. Reading the examples shown for them gives the user an informal idea about the language concepts and the modelling workflow.

Systematic introductions are most efficient (1) for understanding the basic modelling concepts, (2) for finding a starting point for the modelling work and the associated understanding of the modelling workflow, and (3) for acquiring a basic understanding of the syntactical structure of the representation. They can also give an introduction to

	Challenges in learning							
Support for learning	Basic concepts	Where to start	Language concepts	Focussing on right subset	Representation details	Reading the representation	Workflow of modelling tool	Features of modelling tool
Context-sensitive description			○		●	●	○	●
Systematic introduction	●	●			○	●	●	○
Transformation patterns	○	○	○	○	●	●		
Personal trace of seen pages	○		○	○	○		○	○
Personal annotations and bookmarks	○		○	○	○	○	○	○
Global history of use cases	○	●	●	●	○			
Automatic model generation			○	●	●			
Automatic methods for help selection	○		○	○	○			
Expert assistance	●	●	●	●	●	○	●	●
Peer assistance	○	●	○	●	○	○	●	●

● strong support
○ some support

Fig. 2. Mapping of learning challenges to support methods.

the individual features of the modelling tool, which is limited by the reader's memory. However, indexing them by the language elements used permits the reader to return to the corresponding introduction when he/she needs it.

Explaining the modelling task as a *transformation of certain patterns* in the input teaches representation details concerned. It also gives an informal introduction of the corresponding parts of language concepts. Under the assumption that the user can detect the patters in the input, it helps to find a starting point for the modelling and to focus on the right subset of the language features. The detection is helped by the fact that these patterns can be described without using advanced modelling terminology which means that even beginners should easily understand when to apply which pattern, as long as the patterns are defined on a generic level.

Personal traces, annotations and bookmarks are a complementary facility supporting most learning tasks. Their efficiency mostly depends on how the user handles them.

The *global history of use cases* helps the user to find suitable starting points for his modelling. While they will help focussing on the right subset of the language, it is not too likely that the representation details in another use case will match the requirements of the new one directly. This means that the global history provides the first step, to be followed by exploring various representation details using systematic introductions or context-sensitive descriptions.

In the context of the learning process, *automatic models generation* are a two-sided aid. On the one hand, they can relief experienced modellers from manual model creation interactions. On the other hand, false positives by the matching algorithm can confuse beginners. However, IE shows consistent syntax examples and focusses on suitable language elements. False positives, once refuted and possibly discussed with others provide interesting learning insights, perfectly grounded by an example taken from the guideline to be modelled.

Automatic selection of information bits shown to the user promises to stretch the horizon beyond matching syntax elements and keywords in the various descriptions.

Expert assistance is universally applicable, but a scarce resource. *Peer assistance* will more focus on where to start, using the tools, and focussing the learning process. Of course, some of the peers will become experts over time which will blur the distinction between the two groups of advisors.

5.2 Scalability

An important influence factor for rating the value of the described features is the size of the modelling community. For very small communities, expert assistance is more efficient than providing extensive training material. As the community grows, such prepared material becomes indispensable.

As the community grows further, both the global use case history and the forums for peer assistance will need structuring to remain manageable and useful.

The collection of transformation patterns as well as IE configuration files will continually grow with the size of the community, making them more powerful in the process.

5.3 Reducing the Need to Learn

It is important to keep in mind means to obsolete learning, because they complement the above solutions. The most important items are:

- A presentation of the syntax which is favourable for the human reader. Syntax highlighting is universally accepted as a standard aid. Where possible, XML should be shown as nested boxes to intuitively represent the hierarchy and to save space over the native XML syntax.
- An easy to understand workflow which does not provide surprises for the user.
- Homogenous design of the representation. Research on both programming language design and guideline modelling shows that languages with a clear and consistent design are much easier to learn than older ones which contain many exceptions and special cases for historic reasons.

While these items are beyond the scope of this paper, it is important to see their role in raising or lowering the bar for the learner.

6 Conclusion

An important part of knowledge modelling for medical applications is modelling of clinical guidelines and protocols. They are important means to improve the quality of care. However, they need to be translated to computer-interpretable models.

There are few persons who have sufficient skills in both medicine and computer science. Forming teams of experts in both fields increases the organisational difficulty and thus the modelling cost. Therefore, training persons with various backgrounds on the modelling job is an important challenge in medical knowledge modelling.

Various solutions have been proposed for comparable settings. They complement each other and none of them can cover all the requirements. The union of all the solutions can easily overwhelm the user. Therefore, we believe that benefits only exceed user effort when these offerings for support are kept in the background, and if they are fully integrated into the editing environment.

We draw our input from three different sources: teaching medical informatics students, training modellers in various research projects, and literature on similar training initiatives and on teaching programming, which shares important features with knowledge modelling.

Difficulties of learners can be grouped as follows: Understanding the principles of the guideline modelling process; understanding the concepts in the representation used; focussing the learning process on those parts of the knowledge which are needed (first); and handling of the modelling tool used.

With our approach we want to support modellers by a range of complementary subsystems: (1) context-sensitive descriptions of details; (2) systematic discussion of foundations; (3) personalisation through observing one's traces through the knowledge corpus and annotating it as desired; and (4) various communication channels to advising experts and learning peers. Information Extraction techniques increase the efficiency where possible.

The key aspect of our solution is not the novelty of the parts, but the close ties between them. We believe that it is important to provide an integrated combination of all the mentioned features to arrive at a system which optimises the performance of each modeller irrespective of the learning stage. We are not aware of such a system in the literature.

While our research focuses on the field of guideline and protocol modelling and the stakeholders involved, it can clearly be expanded to related fields like ontology modelling and modelling of knowledge from fields other than medicine.

Acknowledgements. This work is supported by "Fonds zur Förderung der wissenschaftlichen Forschung FWF" (Austrian Science Fund), grant TRP71-N23.

References

1. Grimshaw, J.M., Russel, I.: Effect of clinical guidelines on medical practice: a systematic review of rigorous evaluations. Lancet **342**(8883) (1993) 1317–1322
2. Peleg, M., Tu, S.W., Bury, J., Ciccarese, P., Fox, J., Greenes, R.A., Hall, R., Johnson, P.D., Jones, N., Kumar, A., Miksch, S., Quaglini, S., Seyfang, A., Shortliffe, E.H., Stefanelli, M.: Comparing computer-interpretable guideline models: A case-study approach. Journal of the American Medical Informatics Association (JAMIA) **10**(1) (Jan-Feb 2003) 52–68
3. Bonar, J., Soloway, E.: Preprogramming knowledge: A major source of misconceptions in novice programmers. Human-Computer Interaction **1**(2) (1985) 133–161

4. Dagdilelis, V., Satratzemi, M., Evangelidis, G.: Introducing secondary education to algorithms and programming. Education and Information Technology **9**(2) (2004) 159–173
5. Kramer, J.: Is abstraction the key to computing? Commun. ACM **50**(4) (2007) 37–42
6. Freund, S.N., Roberts, E.S.: Thetis: an ANSI C programming environment designed for introductory use. SIGCSE Bull. **28**(1) (March 1996) 300–304
7. Efopoulos, V., Dagdilelis, V., Evangelidis, G., Satratzemi, M.: Wipe: a programming environment for novices. In: Proc. of the 10th annual SIGCSE Conference ITiCSE, New York, NY, USA, ACM (2005) 113–117
8. Shalom, E., Shahar, Y., Taieb-Maimon, M., et al.: A quantitative evaluation of a methodology for collaborative specification of clinical guidelines at multiple representation levels. Journal of Biomedical Informatics **41**(6) (2008) 889–903
9. Shalom, E., Shahar, Y., Taieb-Maimon, M., et al.: Can physicians structure clinical guidelines? experiments with a mark-up-process methodology. In Riaño, D., ed.: Knowledge Management for Health Care Procedures. Springer Berlin / Heidelberg (2009) 67–80
10. Shahar, Y., Young, O., Shalom, E., Galperin, M., Mayaffit, A., Moskovitch, R., Hessing, A.: A framework for a distributed, hybrid, multiple-ontology clinical-guideline library, and automated guideline-support tools. J Biomed Inform **37**(5) (October 2004) 325–344
11. Patel, V.L., Allen, V., Arocha, J., Shortliffe, E.H.: Representing clinical guidelines in GLIF: Individual and collaborative expertise. JAMIA **5**(5) (1998) 467–483
12. Patel, V.L., Branch, T., Wang, D., Peleg, M., Boxwala, A.: Analysis of the process of encoding guidelines: An evaluation of GLIF3. Methods Inf Med **41**(2) (2002) 102–113
13. Peleg, M., Patel, V.L., Snow, V., et al.: Support for guideline development through error classification and constraint checking. In: Proceedings of the AMIA Annual Symposium. (2002) 607–611
14. Aigner, W., Miksch, S.: Communicating the logic of a treatment plan formulated in asbru to domain experts. In: Proc. of the Symposium on Computerized Guidelines and Protocols (CGP 2004). Volume 101., IOS Press (2004) 1–15