Making Means-End-Maps Workable for Recommending Teaching Methods

Michael Koch, Dieter Landes

Faculty of Electrical Engineering and Informatics
University of Applied Sciences and Arts
96450 Coburg, Germany
{michael.koch, dieter.landes}@hs-coburg.de

Abstract. Finding appropriate didactical approaches for a specific purpose in software engineering education is difficult. Our work focusses on a recommendation engine for teaching methods. This encompasses modeling teaching goals and suitable teaching methods. To that end, we translated Reich's pool of domain independent constructive teaching methods into a concept map which also includes educational goals or skills at which these methods aim. We started out from Means-End-Maps (ME-Maps), i.e. simple concept maps based on *i** which aim at modeling goals and tasks to achieve these goals. Modeling Reich's pool of methods revealed several shortcomings of ME-Maps. This article presents experiences we made with ME-Maps, discusses necessary changes and extension, and outlines an editor to create such models. Our extension to ME-Maps is expected to significantly improve readability and overview by providing a visual map to quite complex models. Further, such concept maps establish a basis for a goal-oriented search engine for teaching methods in software engineering education.

Keywords. Means-End-Maps (ME-Maps), Concept Maps, Teaching Methods, Educational Goals, Recommender System, *i**

1 Introduction

Nowadays, it is commonly accepted that successful learning requires advanced teaching methods which reach far beyond traditional instructive formats. In particular, a large variety of active learning methods has been developed over the years in pedagogy. Yet, instructors are experts in their particular domain, say in software engineering, but often lack a profound pedagogical background. Therefore, they need support in choosing appropriate didactical methods for a specific purpose. In order to offer a wider variety of teaching methods, thus enhancing interaction in software engineering lessons, useful didactical methods need to be modeled jointly with goals that they may help to achieve and experiences related to their application in a specific setting.

Yet, it is still an open issue which modeling notation is most appropriate for that purpose, striking the balance between clarity and simplicity on the one hand and sufficient expressive power on the other.

This is why we explore in more detail whether Means-End-Maps (ME-Maps) [9] are appropriate for supporting instructors in a manual search for suitable didactical approaches, but also provide a basis for automated reasoning on promising methods.

Our work is primarily directed towards building a goal-oriented search engine which allows instructors to enter their intent and, in return, provides a set of teaching methods ranked by their suitability to meet these goals in a given context. A crucial component of our work consists in establishing a basis for recommendations on didactical approaches in software engineering education [2, 3]. To that end, we started out to model Reich's pool of constructivist teaching methods [6] with ME-Maps. This pool is an extensive collection of teaching methods which are described explicitly and domain-independently and also pays attention to general prerequisites for a meaningful use of a given method, such as minimum and maximum numbers of participants, timeframe etc. Since the method pool is domain-independent, method descriptions neglect technical outcomes in favor of skills that a specific method will foster.

In the following section, we summarize an adaptation of syntax, semantics, and pragmatics of ME-Maps to make them suit our needs. These adaptations are based on experiences made when modeling Reich's pool of constructivist methods with classical ME-Maps. We also briefly highlight some features of a modeling tool for our variant of ME-Maps before a summary and outlook concludes the paper.

2 Modified Means-End-Maps

2.1 Why Using a Concept Map Based Approach in General?

Concept maps in general are intended to capture domain knowledge by describing concepts and their relationships [5] concisely, thus making the notation fairly intuitive. In our particular context, namely software engineering education, we need to consider teaching methods and their contribution to foster competencies in general. For making good decisions, it is also necessary to pay attention to the instance level, e.g. aspects of the instructor's personal attitude and the contents. Association rules are promising candidates to capture this particular aspect.

Therefore it seems to be a good choice to employ a hybrid recommendation algorithm using concept maps for domain knowledge and association rules for context knowledge.

It is worth noting that we explicitly do not want to build a recommendation engine for the one and only "perfect" method in a specific setting, but providing the instructor with a targeted list of promising methods based on matching the primary and secondary goals as well as the context.

Transparent recommendations are more accepted than non-transparent ones [8]. Thus, for better user acceptance, we also want to make the recommendation process comprehensible to the instructor by being able to explain how and why the system generated a recommendation.

2.2 Introducing a Modified ME-Map Approach Based on Our Experiences

We started out by trying to model Reich's pool of constructivist methods with the strict version of ME-Maps and the recommended CmapTools [1] presented in [9]. The original approach is intended to be minimalistic and comes along with only a few language elements based on *i** [10]. There are just two node types: tasks covering the concept of tasks and hard goals from *i**, and qualities covering the concept of soft goals from *i**, yet in a more focused fashion as quality attributes associated to tasks. To express relationships, achieved-by links, consist-of links, association links and contribution (+, -) links are offered. Here we were confronted with some obstacles regarding to our purposes, leading to some syntactic and semantic modifications, which we will describe and explain in the following.

2.2.1 Task and Method Node

Tasks describe actions performed by participants involved in a specific method.

Methods are a special kind of task which represents teaching methods and plays a central role for our purposes. Methods are derived from the task element and are associated with additional attributes for classification and filtering. For highlighting teaching methods and better distinction from "regular" tasks, their label is printed in bold-face type.

2.2.2 Soft Goal and Quality Nodes

Quality nodes in the original ME-Map approach are intended to cover the concept of soft goals from *i**. They express desired quality attributes associated with tasks [9]. In our point of view, however, these concepts are different: for soft goals, methods are a means to achieve the goal while qualities denote constraints on methods, i.e. a second-order concept. Hence we slightly adjust their appearance to emphasize their characters.

Soft Goals mainly represent competencies or intended outcomes [4] fostered by carrying out a given method or performing a given task. In analogy to the concept of misuse cases [7], we want to be able to express outcomes from contradictive teaching approaches that should be explicitly avoided by inverting their color. This explicit syntactic finesse reduces the effort for sentiment analysis significantly.

Qualities in our definition represent quality attributes that a method or task requires in order to be performed meaningfully. To distinguish qualities from soft goals, their label is printed in bold-face type and in italics.

2.2.3 Generalization Links

The *generalizes* link may be used to model more specialized variants of a task, method, or goal. Derived elements inherit all aspects from their parents and allow for the definition of additional aspects in a specialized context. Thus, replicated parts of the model may be avoided and redundancy be reduced.

2.2.4 Containment Links

The *contains* link express either that a task has multiple sub-tasks or that a goal has multiple sub-goals. In contrast to the *consists-of* link described in [9], the containment relationship may be incomplete. This semantic redefinition was necessary in the context of education and distributed modeling, since it might not be useful to model all sub-competencies of a higher competency if these aspects are not relevant for a particular method, but matter for other methods in another (partial) model.

2.2.5 Achievement Links

The *achievedBy* link is semantically identical to the *achieved-by* link from the original ME-Map approach and is intended to be an equivalent of means-end links in *i**. It indicates tasks respectively methods – which are derived from tasks – offering solutions for a parent task. Sibling tasks respectively methods are alternative means to the end represented by the parent task.

2.2.6 Requirement Links

The *requires* link is used to describe required goals needed to achieve another goal or to carry out a task respectively a method. Since this concept is closely related to the concept of association links between tasks/methods and qualities, we replaced association links from the original ME-map approach by requires links. In contrast to association links, the latter link type is also directed to emphasize the roles.

2.2.7 Contribution Links (+, -)

Positive (+) and *negative (-) contribution* links are used to describe the impact of a task, method, or goal on the acquirement of a competency. These links have an outstanding importance for recommending suitable teaching methods. Since the influence of a given method to the achievement of a given competency is hard to express by a quantitative value, it seems more reasonable to use a qualitative value. This contribution can either be positive or negative, in contrast to *i** with its contribution links break, hurt, some-, some+, help, and make. We also decided to not use a qualitative scale like -- or ++ since this might suggest higher precision, yet might cause vagueness if the criteria leading to a rating are not defined or disputable. A fine-grained qualitative scale would also aggravate the occurrence of semantic conflicts when distributed partial models are merged since it is likely that most instructors classify positive and negative aspects similar but weigh them in a different manner. There is also no unknown or neutral contribution intended, since it would have no effect on the generation of recommendations, but would only increase the complexity of the model at the expense of readability.

2.3 An Example of Our Modified Means-End-Map Approach

Figure 1 shows an exemplary (incomplete) model of an in-tray exercise – which is also popular in job application assessment centers – using our modified notation.

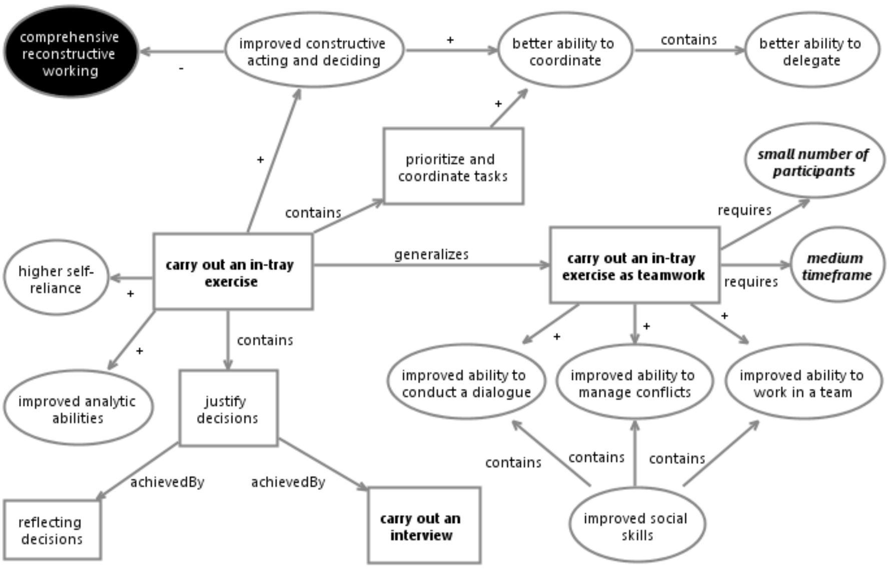
comprehensive reconstructive working
-
improved constructive acting and deciding
+
better ability to coordinate
contains
better ability to delegate
+
prioritize and coordinate tasks
+
small number of participants
contains
requires
carry out an in-tray exercise
generalizes
carry out an in-tray exercise as teamwork
medium timeframe
requires
higher self-reliance
+
+
+
+
contains
+
improved analytic abilities
justify decisions
improved ability to conduct a dialogue
improved ability to manage conflicts
improved ability to work in a team
contains
contains
contains
achievedBy
achievedBy
reflecting decisions
carry out an interview
improved social skills

Fig. 1. Modified Means-End-Map for In-Tray Exercises

3 Tool support

The CmapTools recommended for drawing ME-Maps in [9] has many advantages and allows collaborative modeling. Our adjusted approach can be modeled with this tool. However, the CmapTools lack capabilities for custom extensions, such as defining custom attributes. Hence, we developed an Eclipse plugin offering all tools to draw modified ME-Maps, element-based and attribute-based filters as well as some common features such as, e.g., the ability to add hyperlinks or to store a detailed description for each element that will appear on mouse-over events.

Since distributed modeling is important in our setting to create a joint method pool, all elements are tagged with a content-independent globally unique identifier (GUID). Partial models may be merged by creating an alias for multiple GUIDs. This approach allows us to retain the original models, which facilitates correcting wrong associations done by either an algorithm or an instructor, and it can be used to train the merging and recommendation algorithms by implicitly defining synonyms.

4 Summary and Future Work

Modeling didactical methods and the outcomes that they intend to produce is an important prerequisite for supporting instructors in choosing those methods that best suit their needs. This paper explored whether Means-End-Maps are an appropriate notation for that purpose by modelling Reich's pool of constructivist methods. As it turns out, ME-Maps cannot reasonably be used out-of-the-box since some notational elements do not really fit our needs, both syntactically and semantically. In particular, there is a semantic mismatch between ME-Map's qualities and goals in an educational setting. Therefore, we propose a variation of ME-Maps that seems to be better adapted to the modeling requirements in the educational domain. As future work, this notation will be further explored in the context of an intelligent recommendation system for didactical methods.

Acknowledgements

Our research is supported by the German Ministry of Education and Research (Bundesministerium für Bildung und Forschung) as part of the project EVELIN under grant no. 01PL12022A. For additional information see http://www.evelinprojekt.de.

References

1. Cañas, A. J. et al.: CmapTools: A Knowledge Modeling and Sharing Environment. In: Proc. 1st International Conference on Concept Mapping, Pamplona, Spain (2004)
2. Koch, M., Landes, D.: A Recommender System for Didactical Approaches in Software Engineering Education. In: Proc. DeLFI Workshops 2014 (DeLFI 2014), CEUR Workshop Proceedings Volume 1227, pp. 140-143, Freiburg, Germany (2014)
3. Koch, M., Landes, D.: Design and Implementation of a Competency Repository. In: Rocha, A. et al. (eds.): New Perspectives in Information Systems and Technologies, Volume 1, pp. 249-255, Springer, Heidelberg, Germany (2014)
4. Koch, M., Landes, D.: Notations for Modeling Educational Goal Profiles. In: Proc. 1st European Conference of Software Engineering Education (ECSEE 2014), pp. 45-58, Shaker, Aachen, Germany (2014)
5. Novak, J.D., Cañas, A. J.: The Theory Underlying Concept Maps and How to Construct and Use Them, Technical Report IHMC CmapTools (2008) – available at: http://cmap.ihmc.us/docs/theory-of-concept-maps (last visited on: 2015-06-10)
6. Reich, K.: Konstruktivistische Didaktik. 4th ed., Beltz Verlag, Weinheim, Germany (2008) – The method pool is available in German at: http://methodenpool.uni-koeln.de (last visited on: 2015-06-10)
7. Sindre, G., Opdahl, A. L.: Capturing Security Requirements through Misuse Cases. In: Proc. Norsk Informatikkonferanse (NIK'2001), pp. 219-230, Tromsø, Norway (2001)
8. Swearingen, K., Rashmi, S.: The Role of Transparency in Recommender Systems. In (Terveen, L. G. et al. Eds.): Extended abstracts of the 2002 Conference on Human Factors in Computing Systems (CHI'02), pp. 830-831, ACM press, New York (2002)
9. Wang, J., Sturm, A., Yu, E.: Know-How Mapping: From *i** to ME-maps. In: Proc. 7th International *i** Workshop (istar'14), CEUR Workshop Proceedings Volume 1157, Thessaloniki, Greece (2014)
10. Yu, E.: Modeling Strategic Relationships for Process Reengineering. PhD thesis, Department of Computer Science, University of Toronto (1995)