Identifying HCI Patterns for the Support of Participatory Enterprise Modeling on Multi-touch Tables

Anne Gutschmidt, Valentina Sauer, Kurt Sandkuhl, Alexey Kashevnik

Identifying HCI Patterns for the Support of Participatory Enterprise Modeling on Multi-Touch Tables

Anne Gutschmidt[1], Valentina Sauer[1], Kurt Sandkuhl[1], and Alexey Kashevnik[2]

[1] University of Rostock, Computer Science Department
anne.gutschmidt@uni-rostock.de
[2] ITMO University, Information Technology and Programming Faculty

Abstract. This paper deals with the question of how software enabling participatory enterprise modeling on a multi-touch table should be designed. We will present a pre-selection of existing HCI patterns addressing the requirements which come along with collaboratively creating enterprise models on a shared workspace. Moreover, we examined a software prototype based on a task model and video analysis. The videos show participatory modeling sessions and give hint on frequent activities and deficiencies of the prototype. Based on our results, we will give recommendations of HCI patterns which should be applied when designing software tools for participatory enterprise modeling on multi-touch tables.

Key words: HCI patterns, participatory enterprise modeling, multi-touch table, task analysis, video analysis

1 Introduction

Enterprise models are supposed to capture and represent the situation in an enterprise, either in terms of the current state of affairs or of the planned future situation [1]. In this context, a precondition for high quality enterprise models is to fully and correctly elicit the relevant knowledge from within the enterprise under consideration for the defined scope and purpose of modeling. Participatory enterprise modeling (PEM) is an elicitation technique considered as in particular valuable, when an agreement and a joint view of different stakeholders on the current or future situation are important [2]. Various methods, techniques and tools have been proposed by the scientific community to support PEM (cf. section 2.1). However, constantly emerging new technologies make more and more new tools possible. We argue that with an increased use of multi-touch tables (MTT) and large touch screens, more attention should be paid on adapting or specifically designing tools for participatory, facilitated and collaborative EM. More concretely, the paper addresses the design of user interface and human computer interaction (HCI) for MTT in the context of PEM. This may also contribute to light-weight modelling tools and the research agenda for extending the reach of enterprise modeling [3]. Our conjecture is that HCI patterns from

software engineering (cf. section 2.2) provide relevant and reusable knowledge for the design of PEM tools. Based on a general task analysis for goal modeling as selected part of EM and using the results of a video analysis revealing problems and challenges in PEM on a multi-touch table, we aim at contributing to an understanding of specific requirements in PEM tool design. The main contributions of the paper are (1) a list of HCI patterns supporting participation and enterprise modeling, (2) a task analysis of typical EM activities and (3) results of evaluating the HCI patterns for improving a modeling tool on a multi-touch table. The remainder of the paper is structured as follows: Section 2 will present the theoretical background dealing with the areas of PEM (section 2.1) and HCI patterns (section 2.2). In section 3 we will present our selection of HCI patterns where we list patterns we found most fitting for PEM on a MTT. We examined a software prototype to determine which of the previously selected HCI patterns have already been applied (section 4). Furthermore, we documented basic user interactions enabled by the prototype in a task model (section 5) and, based on video recordings of thirteen PEM sessions, we analyzed the interactions with the aim of identifying potentials of improvement (section 6). The paper closes with a general discussion in section 7.

2 Theoretical Background

2.1 Participatory and Collaborative Enterprise Modelling

A General Background In general terms, EM addresses the systematic analysis and modelling of processes, organization structures, product structures, IT-systems or any other perspective relevant for the modelling purpose [4]. A detailed account of EM approaches is provided in [5]. PEM and involving different stakeholder groups in EM has a long tradition (see, e.g., [5]). Since several stakeholder groups are involved in the modelling process and have to work together on one model, this process calls for participation of everyone involved. In this PEM process the methodology experts and domain experts work together on the model [6]. By working together right from the beginning, it is more likely that the final model will be accepted by the participants and they will commit to it. Furthermore, the stakeholders will agree with the model, after all, they worked on it, too. Another advantage of PEM sessions is that they can increase the quality of the model, by introducing people into the process who hold valuable knowledge of the enterprise and its processes. Domain-specific modelling languages (DSML) [7] are supposed to support these various stakeholders in model creation and use. The scientific literature on EM offers several views as its constituents (see, e.g., [8, 9]), like the modelling procedure or modelling method, the result of modelling (i.e. the model), the tool support, and the organizational structures establishing modelling within an organization. However, not all researchers in EM agree on the above EM constituents. Some researchers emphasize the importance of meta-models and modelling languages for capturing different perspectives [8]. Tool support is often seen as inseparable manifestation

of modelling approaches and notations [10], but in other research work as aid to support modelling [11]. Organizational structures and role descriptions are often neglected in EM approaches.

Participatory Enterprise Modeling Sessions When an enterprise decides to start an enterprise modeling project with actively involving stakeholder representatives, they will have to invest resources into that project: Most obviously, they will have to exempt employees from work to let them take part in modeling sessions. The participants should come from different parts of the company, and have adequate domain knowledge which is why they are called domain experts. They should also have the authority to suggest organizational changes contained in the final models [2]. Stirna and Persson [2] propose a number of 4-8 participants per session. In addition to domain experts, a company should recruit so-called method experts. Their purpose is to support the domain experts in creating enterprise models based on their knowledge of modeling notation and method. Usually, a facilitator leads the discussion and modeling process while being completely neutral about the content. A tool operator assists the domain experts in creating the actual models. He or she helps handling the modeling tool and generating syntactically correct models. Optionally, a secretary may take additional notes to document the rationale of the creation process [1, 2]. An enterprise modeling project may comprise multiple modeling sessions lasting several hours and possibly involving different domain experts who create and refine models [2].

Modeling Language Enterprise models are usually represented by diagrams containing geometric shapes such as rectangles or circles. These shapes reflect concepts and are usually connected by lines or arrows representing relationships. All model elements may be labeled, giving further information. In a formal language both syntactical and semantic rules have to be followed when drawing the actual model [2]. A goal model in the 4EM notation, for example, consists of differently colored rectangles, e.g., a green rectangle represents a goal, an orange rectangle represents a problem. The rectangles usually contain an expressive description and a number. To show that a certain problem hinders a goal, a relationship between these components must be added including the respective label [1].

2.2 HCI Patterns

HCI patterns, also called HCI design patterns, describe successful best practice solutions for reoccurring User Interface (UI) design problems, therefore also affecting implicitly the usability of software tools [12]. These patterns should support the designers and keep them from reinventing the same solutions over and over again. Their advantage is that useful design solutions can be captured and generalized in the form of a pattern to solve similar problems with them [13]. The development of UIs is complex, therefore, reusing knowledge, already gained by previous design processes, helps the designers and developers to work

more efficiently and improve their productivity [14, 15, 16, 17]. A pattern is the relationship between a certain context, problem and solution [18]. It describes the context within which the patterns can be used, the problem that has to be solved by the pattern and its solution [17]. Initially, this idea of patterns was developed by Christopher Alexander for architectural designs [18]. The "Gang of Four" adopted the pattern concept for the design of object-oriented software [19]. Eventually, patterns were also adopted by the HCI community. While the Gang of Four gives instructions about how to implement a pattern, HCI patterns are about the general design of an interface and its purpose for the user. The pattern concept not only included the patterns themselves, but also a pattern language. A pattern language consists of patterns and their relationships, i.e. a network of patterns. High-level patterns in this network may be solved by low-level patterns [16]. Since the patterns of one language are connected to each other, it is apparent that a pattern language combines patterns for a given family of design problems in a specific domain [15, 20]. Successor and predecessor relationships between patterns are a key concept when working with pattern languages, since they enable finding closely related patterns [15].

3 Selecting HCI Patterns for Participatory Enterprise Modeling on MTTs

We have scanned existing lists of patterns [21, 22, 23, 24, 25] and further works presenting HCI patterns [26, 27] which covered concepts that could be applied to MTT. While the lists of Tidwell [23, 22] and van Welie ([21] are most often cited, Remy et al. [24] created a pattern list specifically for the MTT. We particularly looked for HCI patterns that fit the requirements of the special context of PEM with a multi-touch table. We formulated major concepts which helped us selecting and categorizing fitting patterns, and also reflected the above-mentioned requirements. Figure 1 shows these concepts in bold letters with thick frames at the top of the diagram. The remaining elements represent existing HCI patterns we have found in the above-mentioned sources. The arrows represent relationships among the elements, e.g. space may be saved using collapsible panels. A pattern may also serve several concepts. Moreover, patterns may be related.

Usually, enterprise models become very big and complex. So, space for interactions will become more and more scarce as a model is growing. To **save space**, several patterns may be used, such as *collapsible panels* or *hover tools* [23]. Tidwell introduced the pattern *hover tools* for mouse-based applications [23], where elements are displayed only when hovering the mouse icon over an object. For touch devices, there is not yet an equivalent to hovering, but only touching. Tidwell is of the opinion that touching may cause precipitate commitment. Nevertheless, it may ensure that the displayed model is not cluttered by displaying editing options which are not needed at the moment. Different *views* [21] may be used as an alternative, where users may switch between editing view and "final" view that is showing just the model.

Depending on the size of the table and of the model elements, it may be difficult to see/read or reach certain objects. As mentioned before, there should be 4-8 domain experts plus at least one method expert present at a modeling session. Thus, the software must present the model in a way that is visually and physically available to several persons at a time. Patterns such as *zooming* [24, 23] and *extending reachability* [24] support **physical and visual reachability**.

As mentioned above, the modeling tool is mainly handled by the tool operator. However, Stirna and Persson suggest that domain experts should be involved by e.g. letting them write down their ideas on colored cards, present them to the group, discuss them and then cluster related cards [2]. Thus, the editing software for the MTT should not be tailored to only the tool operator. It should also offer domain experts an easy way of capturing their ideas in their own words with the MTT. A third party like the tool operator may accidentally change the meaning of statements. Still, the tool operator may then assist in composing a syntactically correct model. Furthermore, the software should not be designed in a way that one person may take over a whole modeling session. Remy et al. [24] introduced a pattern called ***balanced participation***. This implies that there must not be any conflict about or restricted access to resources, especially input devices. An *overlay menu* [21], possibly with multiple instances, instead of a single fixed menu could support this. When providing a horizontal work surface, different perspectives must be provided for users possibly standing at all sides of the table. This is addressed by the pattern *desktop orientation* [24] meaning that the orientation of the interface can be changed. Balanced participation could also be promoted by *user identification* [24]. In *private spaces* participants may take notes of their own ideas, possibly with *embedded electronic devices* such as tablets [24], before sharing them with the group analogous to the above mentioned card writing. However, the content produced in private space should be meant to be shared, otherwise it might undermine collaboration.

The **modeling** task itself brings some special requirements with it. As mentioned before, models can become very complex. So, the table should be large enough to both display the model and let all participants have access to the model (*large collaboration table* [24]). According to [2], domain experts should not be burdened with details of the modeling notation. Consequently, at best, the software should make obvious what can be done (e.g. with *input hints* [23]), and it should not allow what should not be done (e.g. with *constraint input* [21]), possibly already considering notation rules.

As domain experts should not be expected to be experts on digital touch devices such as tabletops either, the software must be very **intuitive** and **easy to handle**. An intuitive interface may be implemented based on patterns such as *input hints*, *good defaults* [23] and *constraint input* [21] such that users know what to do. With easy handling, we mean that it should not be difficult or effortful to see content on the table or to perform an interaction on the MTT. E.g., the "fat-finger" problem may be prevented by applying a pattern such as *generous borders* applied to the components of a model or buttons and keys

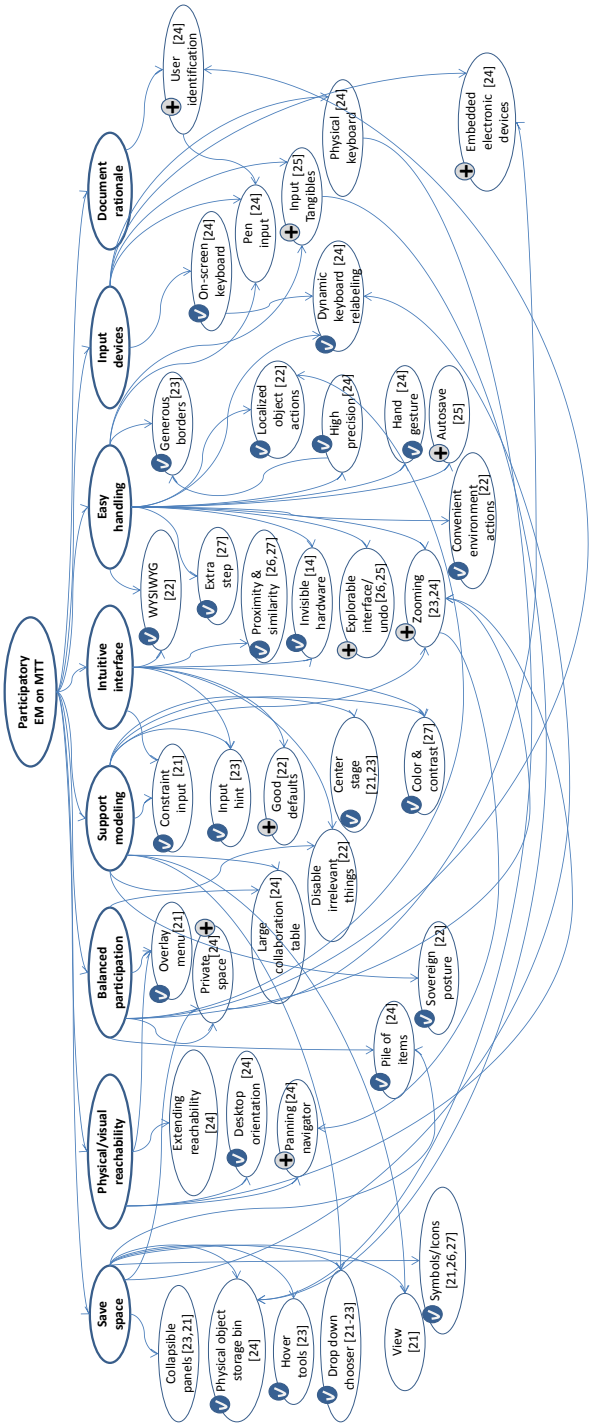

Fig. 1. Selection of HCI patterns suitable for PEM on a MTT. Please note that HCI patterns may have different names in the pattern catalogues. A check mark means, the pattern has been found in the software, a plus mark means, the pattern should be added.

[23]. The pattern *WYSIWYG* (what you see is what you get) [22] should make interactions quicker, as immediate feedback of one's action is given.

With the MTT, different **input devices** are available. While the *physical keyboard* is often felt as more convenient, but occupying space on the work surface, *on-screen keyboards* may be instantiated for every user at each required spot and easily dismissed if no longer needed [24]. *Input tangibles* may be used as an alternative [24], although there must be some additional space where these objects can be stored beyond the work surface (*physical object storage bin* [24]).

In an enterprise modeling session, it is also of interest how ideas evolved. The **rationale** may be documented by a secretary [1]. *User identification* may add information in a way that the author information of components in the model can be saved in addition.

4 Identifying HCI Patterns in a Prototype PEM Editor

In order to confirm the suggested HCI patterns, they should actually be applied in existing software. To our knowledge, there does not yet exist a commercial enterprise modeling editor especially developed for collaborative working with a MTT. Therefore, we have examined a prototype developed at the university of Rostock, as a starting point. In previous studies, we have worked with this prototype [28, 29] which allows creating goal models according to the 4EM notation on a MTT. In particular, it supports collaboration by enabling simultaneous input by several users. We wanted to know whether some of our selected HCI patterns from section 3 have already been applied in the software and present their concrete implementation. Due to space limitations, we can only describe a small selection. In Fig. 1 we have marked the patterns we have found in the prototype with a check mark.

In the editor, *localized actions* [22] in terms of buttons directly accompanying components and relations, simplify the handling and support *balanced participation*, i.e. users can manipulate all the objects they can reach without having to access a menu possibly situated somewhere else. E.g., each component has a button to set it to an editing mode and to generate a new relation starting from this component. Moreover, when the user touches one of the text fields of a component which is in editing mode, an *on-screen keyboard* [24] is attached right below the component. This keyboard belongs only to this component (*localized object actions*), every component may have its own keyboard. Thus, the keyboard is not a resource to be shared which should also promote *balanced participation*. Thus, actions referring to an object are situated in its close proximity as can be seen in Fig. 2a.

These buttons, however, are hidden by default in order to save space and keep an uncluttered view. Only when a user touches the component, the buttons appear. After a few seconds, the buttons slowly fade out following the pattern *hover tools* [23] (see Fig. 2a).

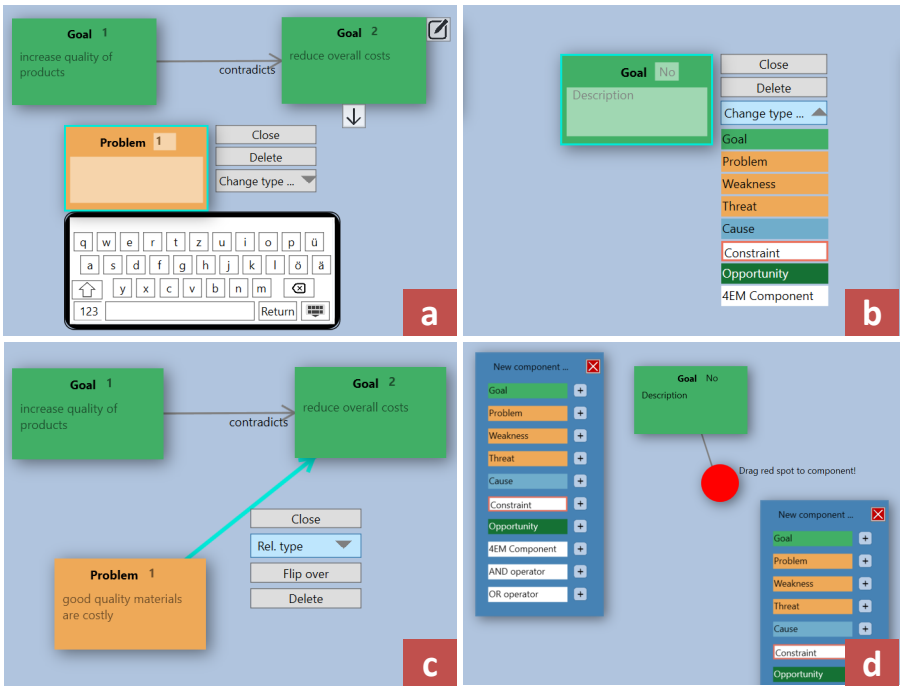

Fig. 2. Example screenshots of the prototype showing a) localized object actions, on-screen keyboard and hover tools, b) drop-down chooser for selecting a component type, c) a relation in editing mode with drop-down chooser, and d) overlay menus and a newly created relation with tool tip/input hint.

By offering the possibility to rotate components, the pattern *desktop orientation* [24] is partly implemented. Only single elements, but not the whole model can be rotated to a participant's respective orientation.

When a user wants to create a new component, a menu (see Fig. 2d) must be opened by *hand gesture* [24], namely tap and hold. The same gesture is also used to set a relation into editing mode, e.g. for setting a label or deleting it. There is no fixed menu, but the menu can be opened at any point on the work surface as described in the pattern *overlay menu* [21]. The pattern *balanced participation* [24] is implemented by allowing several instances of the menu. That way, participants do not have to share this resource. For the creation of the actual component from the menu, the pattern *constrained input* was used. The menu allows the creation of only those elements that are included in the modeling language. There is no free drawing.

WYSIWYG [23] is applied when drawing a relation and moving elements. E.g., components may be moved and rotated, and the effect of these actions can be seen immediately. Moreover, if a relation is connected to a component in movement, the relation's orientation and length is adapted automatically like a physical rubber band.

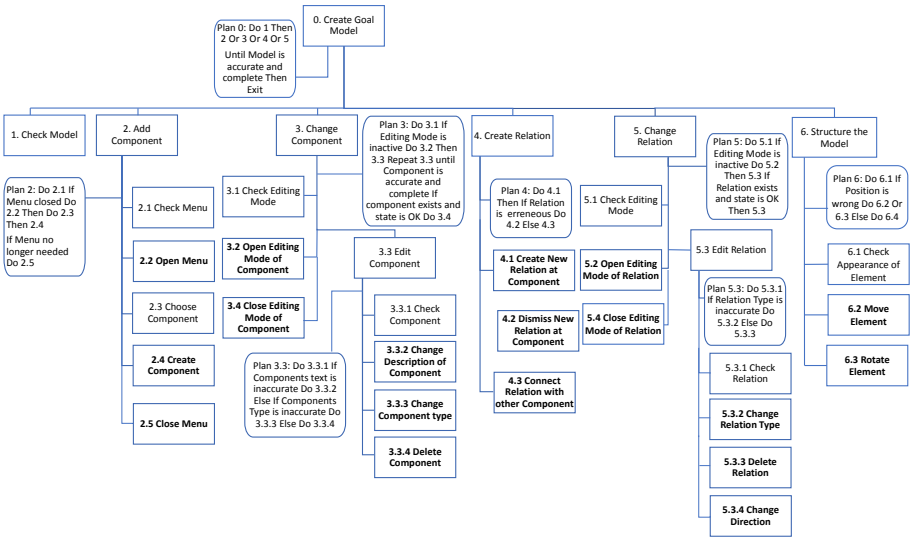

Fig. 3. Task model with basic user interactions with the prototype modeling editor.

Although the physical conditions do not belong to the software, we want to add some more patterns which may also have an influence on using it. The multi-touch device is embedded in a wooden table making the *hardware invisible* [14] and offering some space on the table's wooden frame and below the table to store physical objects such as a physical keyboard or handouts (*physical object storage bin* [24]). Due to this setting, we are dealing with a horizontal work surface which cannot be *tilted* [24]. It was technically not possible to use *tangible objects* nor *user identification* [24] with the concrete device (cf. section 6.1).

5 Task Analysis

In order to further examine the software prototype and find potentials of improvement, we wanted to create an overview of user interactions with the software necessary to generate a model. We decided to use Hierarchical Task Analysis (HTA) to attain a graphical representation of theses interactions we could then examine. HTA may help discovering those parts of a task which may cause a user to eventually fail or to succeed [30]. The basic idea of HTA is that there is a general task at the highest level which consists of an operation. Each operation is connected with a goal whose accomplishment can be measured. Goals can be decomposed into sub-goals, thus the connected goals are decomposed into sub-goals. So-called plans determine the order in which (sub-)operations should be executed, including the formulation of conditions and circumstances by which operations are triggered [30, 31]. Examining the software prototype, we considered creating a goal model as main operation which we decomposed

into sub-operations. We furthermore defined plans indicating when each operation is triggered. The resulting task model (see Fig. 3) will lead us later in the observation study presented in section 6.

When creating a goal model with the prototype editor, the user may repeatedly check the model before deciding on an action. When the user decides to add a new component, a menu must be opened offering the possibility to create as many components as desired. When the user wants to change the description or type of a component, or wants to delete the component, the editing mode of the component has to be started. Relations between components may be created starting from one component, drawing the relation to the target component. If a relation was created erroneously, it may be deleted right away. Relations may be labeled with a type by first starting an editing mode. The editing mode is also necessary if the relation is to be deleted. The general appearance of the model may be changed by moving and rotating components.

6 Video Analysis

From a previous study, we used secondary data to find out which of the interactions contained in the task model occur most frequently. We examined video recordings of thirteen modeling sessions performed with the software prototype without interventions of a tool operator. This might give hint on critical points that should be improved or supported in a better way by HCI patterns. For the same purpose, we examined what caused the most difficulties for its users. We also recommend additional patterns, marked with a plus sign in Fig. 1.

6.1 Method and Sample

Thirteen teams of three persons performed an enterprise modeling task on an MTT (3M Multi-Touch Display C5567PW, size: 1210 x 680mm) in a study conducted in 2018 at the University of Rostock [32]. The teams had to create a goal model for a fictitious company within half an hour. 27 of the 39 participants were students, among them students of psychology, business information systems, pedagogy, biology, physics, chemistry, economics, engineering and computer science. On a scale from 1 (novice) to 5 (expert) the participants reported to be quite inexperienced in the modeling notation ($\mu = 1.3, \sigma = 0.8$) and with MTT ($\mu = 1.2, \sigma = 0.5$).

The modeling sessions were video recorded from two perspectives, one showing the table from above, another one capturing the front view on the team. We analyzed the video recordings looking for specific difficulties the participants had during the modeling. Sometimes, participants commented on their problems during the critical incident. At other times, clearly identifying problems with the use of the software turned out to be difficult and is dependent on the observer's interpretation. Moreover, we counted the interactions introduced in our task model (Fig. 3).

6.2 Results and Recommendations

The difficulty which occurred most often ($\mu = 11.2$ times over all sessions) was that menus were opened accidentally. When movements, such as dragging a component, were performed too slowly, this was misinterpreted by the software as tap-and-hold gesture, and unwanted menus were opened. The challenge is to choose hand gestures which are easy enough for the user to perform but clear enough for the system to be distinguished from other actions. We suggest a double tap as a substitute since more complex gestures might make the software less intuitive [24]. Hand gestures have to be thoroughly tested.

Three types of negative incidents were often caused by a lack of space: the editing mode was opened accidentally ($\mu = 3$), a new relation was created by accident ($\mu = 2.6$), and the wrong component was moved ($\mu = 2.3$). As the models grew in complexity, more and more elements were overlapping. To save space, hidden buttons were used. Although the buttons were no longer visible they were still active. This caused users to accidentally press hidden buttons of closely situated components creating new relations etc. To solve this problem, we could disable buttons when they are not visible. Putting buttons inside the components bears the danger of accidentally triggering actions where components should only be moved. A hand gesture could be used to replace one button, possibly mitigating the problem. Close proximity of model elements was, however, only one reason for these negative incidents. Accidental actions were also triggered by participants leaning or putting sheets of paper on the table. Remy et al. [24] suggested *physical object storage bin* for storing items such as keyboard or tangible objects, but no surface to actually lean on or put down sheets of paper is mentioned. In the future, additional frames around the MTT might turn out as a pattern. Nevertheless, such a frame can be in conflict with reachability of all elements on the MTT depending on the size of table and frame.

Considering the interactions to be performed with the goal of **creating an enterprise model**, **opening a component's editing mode** was performed most frequently ($\mu = 31.4$ times over all sessions), followed by **closing a component's editing mode** ($\mu = 30.1$) and **editing a component's description** ($\mu = 29.8$). This frequency might encourage to believe that these interactions should be additionally supported. E.g., one could think about simplifying the access to editing functions as we have already described above. Another option could be to automatically set a component into editing mode after having created it. This, however, could be in conflict with creating a *pile of items* where participants create a kind of repository similar to a stack of cards.

Creating a new relation is the next most frequently performed interaction ($\mu = 25.8$). It is performed by tapping on the component resulting in the display of an arrow button. When the button is pressed a new relation arrow is generated pointing to a red circle that must be drawn to the target component (see Fig. 2b). We observed that some users wanted to draw the arrow button to the target component right away. It seems that this button implies this functionality. Either the button symbol has to be changed or, better because reducing the number

of steps, the expected behavior should be implemented. The latter would also make dismissing new relations obsolete and simplify the creation process.

Drawing a relation occurred 20.8 times on average over all sessions. It seems to work well for the participants, probably being very intuitive. **Opening the editing mode** of the relation was performed equally often ($\mu = 19.8$). The low occurrence of editing interactions such as **changing the relation type** ($\mu = 11.4$) and changing a relation arrow's direction might be explained by the way they had to be accessed (see Fig. 2c for illustration of the editing mode). We observed that some participants did not expect a tap-and-hold gesture but simply tapped once on the relation. The latter would, however, increase the danger of triggering unwanted actions. Nevertheless, it should be taken care that hand gestures are consistent for similar functions. Moreover, *good defaults* could be provided for new relations taking into consideration syntactic rules.

Creating components occurred with an average frequency of 18.8 times. We see the possibility of opening multiple menus for the creation of components at every spot as a major advantage when supporting this interaction. It also enables the creation of a pile of components compared to a participant grabbing a pile of cards he or she can write on.

Closing a menu ($\mu = 14.8$) and **closing the relation editing mode** ($\mu = 14.5$) occurred with a similar frequency. They could be made obsolete by closing them automatically after an interaction was performed. It must be investigated and compared how useful users find each feature.

Deleting a relation ($\mu = 4.6$), **deleting a component** ($\mu = 2.5$), **dismissing a new relation** ($\mu = 2.5$) and **changing a component's type** ($\mu = 2.4$) occurred rarely. Participants did not seem to experiment with model elements after they had created them. Nevertheless, an *undo function* is fundamentally advisable with inexperienced users. **Opening a menu** was also a rare interaction ($\mu = 4.4$). Menus remained open although they took a lot of space. Either users prefer a constantly present menu or the actions necessary to open (tap-and-hold) and close a menu are considered as too effortful.

We also observed how much participants **moved and rotated components**. For the interactions, we measured the average overall amount of time over all sessions. As some users tended to perform one big movement in several small steps, frequencies would have given a distorted impression of the actual movement behavior. We noted that rotation was rarely used ($\mu = 4.6$ seconds). One reason might be that rotating components is too difficult. Secondly, rotating single elements might not be seen as beneficial when the remaining model keeps its original orientation. Remy et al. [24] suggest a generally adaptable desktop orientation. This would be a global function requiring the awareness and approval of all users such that no one will be disturbed while working. 5.8 seconds were spent on average on handing over components to another person, and 13.8 seconds were spent on average on moving components to oneself. Due to space problems and layered objects, users often moved components to a place were they could interact with them more conveniently ($\mu = 17.5$ seconds). Movement that we could not assign to any of the above categories made about 179.1 seconds on

average. We often observed that participants repeatedly rearranged components to minimal extend, similar to fidgeting with a pen.

In one of the modeling sessions, a software bug made the system crash. As there was no *autosave*, the model had to be recreated quickly. Although the bug has been removed, *autosave* is fundamentally advisable.

The space problem caused several difficulties. It could generally be mitigated by a *zoom* function. A global zoom is again a function whose activation must be agreed on by all active users. A *panning navigator* should additionally be used to give users some orientation about what part of the model they are currently viewing. Another option would be to make all elements smaller by default, but still recognizable, and offer a zooming function for a single component for further examination and editing. Furthermore, the menus are very big in relation to the work surface and the other model elements. To save space, the menu could be replaced by simply creating default components, set into editing mode from the beginning. This would, however, make creating a pile of components difficult. The work surface could be extended using *embedded electronic devices* which may also serve as *private spaces*. Finally, one could also consider buying a bigger table, however a *large collaboration table* could undermine reachability.

7 General Discussion

New digital devices such as MTT appear very attractive in the context of PEM. They can be a useful tool for collaboratively gathering knowledge and ideas. The intent of this paper was to present experiences and give inspiration on how to design software for MTT serving PEM. HCI patterns provide proven solutions to frequent design problems which may be reused by interface designers. We have searched existing lists of HCI patterns, many of them do not originally refer to touch applications. We presented a selection of HCI patterns we assume to be suitable for PEM on MTT. However, a pattern is really a pattern when it is repeatedly used. To our knowledge, there is no commercial PEM software which is originally made for MTT. So, as a starting point, we investigated a software prototype to check whether we would find some of the previously selected patterns and we showed what kind of interactions are required to create an enterprise model on an MTT with this prototype. Our task and video analysis have shown that the number of interactions may actually be reduced in the prototype. The results of the video analysis also revealed shortcomings of the prototype which might be overcome by using additional HCI patterns from our selection.

One of our major findings is that certain patterns may be in conflict. E.g., in the prototype, multiple instances of menus and on-screen keyboards were used. On the one hand, this supports balanced participation. On the other hand it takes a lot of space. The lack of space is a severe challenge, yet, a large collaboration table could make it difficult for users to recognize and reach all elements. A zoom function could also help solving the space problem, however, as a global function it might disturb users in their work. Thus, we recommend to use global functions with care. Hand gestures are a beneficial means for saving

space. Nevertheless, we recommend to test which gestures users find intuitive and convenient. Furthermore, there must be consistent gestures for similar functions. We observed modeling sessions where participants were usually standing. We found that some persons tended to lean on the table or put down paper on it. Thus, we would recommend to use a frame around the table, but thoroughly considering that this will not restrict reachability.

Eventually, our selection of HCI patterns can certainly not be considered as complete or final. We hope to be able to investigate more applications in this area in the future to further test, confirm and adapt our selection of HCI patterns.

Acknowledgements. Part of the research has been developed in scope of a project financed by Government of Russian Federation (Grant 08-08).

References

1. Sandkuhl, K., Stirna, J., Persson, A., Wißotzki, M.: Enterprise modeling. The Enterprise Engineering Series. Springer Berlin Heidelberg (2014)
2. Stirna, J., Persson, A.: Enterprise Modeling - Facilitating the Process and the People. Springer International Publishing (2018)
3. Sandkuhl, K., Fill, H.G., Hoppenbrouwers, S., Krogstie, J., Matthes, F., Opdahl, A., Schwabe, G., Uludag, Ö., Winter, R.: From expert discipline to common practice: A vision and research agenda for extending the reach of enterprise modeling. Business & Information Systems Engineering **60**(1) (Feb 2018) 69–80
4. Vernadat, F.: Enterprise modeling and integration (emi): Current status and research perspectives. Annual Reviews in Control **26**(1) (2002) 15 – 25
5. Stirna, J., Persson, A., Sandkuhl, K.: Participative enterprise modeling: experiences and recommendations. In Krogstie, J., Opdahl, A., Sindre, G., eds.: Advanced information systems engineering. Number 4495 in Lecture Notes in Computer Science. Springer (2007)
6. Gutschmidt, A., Sandkuhl, K., Borchardt, U.: Multi-touch table or plastic wall? Design of a study for the comparison of media in modeling. In: Lecture Notes in Business Information Processing. Volume 263. 123–135
7. Van Deursen, A., Klint, P., Visser, J.: Domain-specific languages: An annotated bibliography. ACM Sigplan Notices **35**(6) (2000) 26–36
8. Frank, U.: Multilevel modeling. Business & Information Systems Engineering **6**(6) (Dec 2014) 319–337
9. Henderson-Sellers, B., Ralyté, J., Ågerfalk, P.J., Rossi, M.: Situational method engineering. Springer (2014)
10. Gonzalez-Perez, C., Henderson-Sellers, B.: Metamodelling for software engineering. Wiley Publishing (2008)
11. Dietz, J.: Enterprise Ontology: Theory and Methodology. Springer Berlin Heidelberg (2006)
12. Specker, M., Wentzlaff, I.: Exploring usability needs by human-computer interaction patterns. In Winckler, M., Johnson, H., Palanque, P., eds.: Task Models and Diagrams for User Interface Design. Volume 4849 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2007) 254–260
13. Wurhofer, D., Obrist, M., Beck, E., Tscheligi, M.: Introducing a comprehensive quality criteria framework for validating patterns. In Dini, P., ed.: Computation world, IEEE (2009) 242–247

14. Borchers, J.O.: A pattern approach to interaction design. **15** (2001) 359–376
15. Kruschitz, C., Hitz, M.: Human-computer interaction design patterns. **3** (2010)
16. Kruschitz, C., Hitz, M.: Analyzing the hci design pattern variety. In Hanyuda, E., ed.: Proceedings of the 1st Asian Conference on Pattern Languages of Programs, ACM (2010) 1
17. Guerrero-García, J., González-Calleros, J.M., González-Monfil, A., Pinto, D.: A method to align user interface to workflow allocation patterns. In González Calleros, J.M., Collazos Ordoñez, C.A., Guerrero-García, J., eds.: Proceedings of the XVIII International Conference on Human Computer Interaction. ICPS, ACM (2007) 1–8
18. Alexander, C.: A pattern language. Volume 2 of Center for Environmental Structure series. Oxford Univ. Pr (1977)
19. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design patterns. 39. printing edn. Addison-Wesley professional computing series. Addison-Wesley (2011)
20. Lukosch, S., Schümmer, T.: Communicating design knowledge with groupware technology patterns. In de Vreede, G.J., ed.: Groupware: Design, Implementation and Use. Volume 3198 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2004) 223–237
21. van Welie, M.: Welie.com: Patterns in interaction design. `http://www.welie.com/patterns/index.php` (2008) Accessed: 2019-07-30.
22. Tidwell, J.: Common ground. `http://www.mit.edu/~jtidwell/common_ground_onefile.html` (1999) Accessed: 2019-05-02.
23. Tidwell, J.: Designing interfaces. 2nd ed. edn. Safari Tech Books Online. O'Reilly (2011)
24. Remy, C., Weiss, M., Ziefle, M., Borchers, J.: A pattern language for interactive tabletops in collaborative workspaces. In: Proceedings of the 15th European Conference on Pattern Languages of Programs. EuroPLoP '10 (2010) 9:1–9:48
25. Laakso, S.A.: User interface design patterns. `https://www.cs.helsinki.fi/u/salaakso/patterns/index.html` (2003) Accessed: 2019-05-02.
26. Coram, T., Lee, J.: Experiences - a pattern language for user interface design. `http://www.maplefish.com/todd/papers/Experiences.html#Interaction` (2016)
27. Lockton, D., Harrison, D., Stanton, N.A.: Exploring design patterns for sustainable behaviour. The Design Journal **16**(4) (2013) 431–459
28. Gutschmidt, A.: Empirical insights into the appraisal of tool support for participative enterprise modeling. In: Proceedings of the 9th International Workshop on Enterprise Modeling and Information Systems Architectures, Rostock, Germany, May 24th - 25th, 2018. (2018) 70–74
29. Gutschmidt, A.: On the influence of tools on collaboration in participative enterprise modeling–an experimental comparison between whiteboard and multi-touch table. In: ISD 2018. (2018)
30. Stanton, N.A.: Hierarchical task analysis. **37** (2006) 55–79 Journal Article Research Support, Non-U.S. Gov't Review.
31. Annett, J.: Hierarchical task analysis. In: The Handbook of Task Analysis for Human-Computer Interaction. CRC Press (2003) 83–98
32. Gutschmidt, A., Sauer, V., Schönwälder, M., Szilagyi, T.: Researching participatory modeling sessions: An experimental study on the influence of evaluation potential and the opportunity to draw oneself. In Pańkowska, M., Sandkuhl, K., eds.: Perspectives in Business Informatics Research, Cham, Springer International Publishing (2019) 44–58