Gesture-based Process Modeling Using Multi-Touch Devices

Jens Kolb, Benjamin Rudner, Manfred Reichert
University of Ulm, Germany

Abstract

Contemporary business process modeling tools provide menu-based user interfaces for defining and visualizing process models. Such menu-based interactions have been optimized for applications running on desktop computers, but are limited regarding their use on multi-touch devices. At the same time, the widespread use of mobile devices in daily business life as well as their multi-touch capabilities offer promising perspectives for intuitively defining and changing business process models. Additionally, multi-touch tables will foster collaborative business process modeling based on natural as well as intuitive gestures and interactions. This paper presents the results of an experiment that investigated the way users define and change business process models using multi-touch devices. Based on experiment results, a core gesture set is designed enabling the easy definition and change of business process models with multi-touch devices. Finally, a proof-of-concept implementation of this core gesture set is presented. Overall, gesture-based process modeling and multi-touch devices will foster new ways of (collaborative) business process modeling.

Keywords: Business Process Modeling, Gestures, User-centered Process Modeling, Multi-Touch Application, Process Change Patterns

INTRODUCTION

During the last years multi-touch devices have been increasingly used for realizing business applications, and have become more and more useful for the daily work of business people [1, 2]. Multi-touch devices used in companies range from smartphones to tablets to multi-touch tables and walls. Obviously, screen size affects both mobility and application areas. While smartphones are primarily used for mobile communication (e.g., e-mailing and messaging), tablets and multi-touch tables are applied in the context of collaborative tasks (e.g., joint editing of a business document). In particular, multi-touch devices can be applied to business process modeling as well, i.e., to capture and model business processes while interviewing process

participants. In turn, multi-touch devices having larger screens (e.g., multi-touch tables) can be used to collaboratively model and change business processes [3].

Traditional business process modeling tools have not been designed with multi-touch devices in mind, and hence do not take the specific properties (e.g., small screen size) and interaction possibilities (e.g., gesture-based interaction) of these devices into account [4–6]. Hence, research is required that enables the optimized use of multi-touch devices for business process modeling.

This paper introduces a core set of gestures for business process modeling on multi-touch devices. This core gesture set enables process designers to define and change business process models in an intuitive and comprehensible manner. The paper focuses on the introduction and implementation of the core gesture set and less on the design of a concrete user interface. The core gesture set we suggest is applicable to all screen sizes independent of the multi-touch device used. Furthermore, we exemplarily implemented the gesture set for Apple iPad in the context of the *proView* project (cf. www.dbis.info/proView), which aims at human-centric business process management [7–10]. In particular, *proView* provides techniques that enable personalized visualizations of and interactions with process models based on *updatable process views* [11, 12]. Furthermore, intuitive ways of displaying process models are provided, e.g., diagrams, forms, and trees [13]. Gesture-based process modeling complements this work with sophisticated and intuitive concepts for interacting with process models using multi-touch devices.

Note that this paper significantly extends previous work we presented in [14]. First, we provide by far more technical details, e.g., regarding the design and execution of the experiment identifying the gestures required for business process modeling. Second, we now provide an advanced proof-of-concept prototype of a gesture-based process modeling tool. In particular, this implementation utilizes results of the experiment we conducted. Third, we show how gesture-based process modeling is integrated in the *proView* framework.

The remainder of this paper is structured as follows: The background section gives fundamental insights into multi-touch applications and introduces the process modeling functions to be supported by the gesture set. The following section presents the results of an experiment we conducted to understand how users interact with multi-touch devices and which kind of gestures they prefer when modeling and changing processes. These experimental results provide the foundation of the core gesture set introduced in the subsequent section. The section on *proView* describes how gesture-based process modeling is integrated in our overall process modeling and

visualization framework. A proof-of-concept prototype is presented in the implementation section, while the related work section discusses other approaches targeting at human-centric business process management. The paper concludes with a summary and an outlook.

BACKGROUND

This section describes background information needed for understanding this paper. First, general interaction concepts are introduced, which may be used to interact with an application running on a multi-touch device. Following this, common characteristics of multi-touch applications are discussed. Finally, a core function set required for defining and changing business process models is presented.

Multi-Touch Interaction Concepts

Generally, users may interact in different ways with a multi-touch application.

First, interaction concepts suggested in the context of conventional desktop applications may be applied to multi-touch applications as well; e.g., *menu-based interactions* rely on menus and toolbars to provide available functions to users. As an advantage, this concept allows users to easily *explore* the application when searching for a specific function. However, displaying menus and toolbars requires space on the screen, which is limited on devices with small screens (e.g., smartphones and tablets). Hence, menu-based interaction should only be used for multi-touch applications running on larger screens [15].

Second, *gesture-based interaction* relies on gestures for selecting and using the functions provided by multi-touch application. Thereby, a *gesture* constitutes a movement of one or more fingers on the screen of the multi-touch device [16, 17]. Such a movement is then recognized and interpreted by the multi-touch application and operation system respectively. For example, in a slideshow comprising several pictures, the s*wipe gesture* (i.e., swiping with one finger from the right to the left) can be used to display the next picture. Generally, gestures can be categorized into *physical* and *symbolic gestures* [18]. *Physical gestures* manipulate the virtual objects on the screen directly, e.g., by dragging a virtual element on the screen. In turn, s*ymbolic gestures* are related to the function to be executed (e.g., drawing a plus to add an object). As opposed to *menu-based interactions*, which require space to display menus or other graphical elements, gestures do not require any space on the screen. However, as a barrier for gesture-based interactions with a multi-touch application, users do not always know which functions are supported by it and based on which gestures these functions can be selected.

The latter is especially crucial for novices and unexperienced users of the multi-touch application. Usually, this problem is addressed through a tutorial provided the first time the user starts the application.

Third, a *hybrid interaction* comprising both menu- and gesture-based interactions can be realized. For example, a menu bar may be displayed at the top or bottom of the screen offering the most important functions. Additionally, well-known gestures, (e.g., swipe gesture) may be supported.

Characteristics of Multi-Touch Applications

Besides their gesture-based interaction concepts, multi-touch applications or—to be more precise—their multi-touch capabilities can be described through seven fundamental characteristics.

C1 (Screen Occlusion): The user interface design of a multi-touch application should consider that users may cover screen areas with their hands [15]. For example, dragging an object from the top-left corner of a tablet to its bottom-right corner will constitute a difficult task if large parts of the screen are covered by the user's hand.

C2 (Handling Precision): Conventional desktop applications require a mouse pointer to interact with them. This pointer scales with the screen resolution to ensure high precision of user interactions. Regarding multi-touch applications, in turn, the 'interaction device' is the user's finger [19]. As opposed to the mouse pointer, a finger does not scale up or down when changing screen size. Accordingly, the handling precision of a multi-touch application changes with the size of an object the user has to touch. Studies have shown that an object should have a size of 11.52 mm or more to have a hit probability of at least 95% [20]. Regarding multi-touch process modeling, therefore, it will be crucial to achieve a high handling precision.

C3 (Fatigue of Extremities): When using a computer mouse, the hand does not move a lot. Usually, the arm lays on the table and the mouse is moved with the fingers to reach the corners of the screen with the mouse pointer. In turn, interacting with multi-touch applications frequently requires the movement of the whole arm. Further, the arm does not lay on a table, but has to be hovered in the air during the interaction with the multi-touch application [21]. Obviously, the degree of movement strongly depends on the respective screen size.

C4 (Number of Supported Fingers): Compared to conventional desktop applications using a mouse pointer, a multi-touch application typically supports more than one touch point. Generally, the maximum of touch points an application may support is limited through the underlying hardware and op-

eration system respectively. For example, Apples iPad is able to distinguish between 11 touch points (i.e., fingers) [22]. While this number is sufficient for single user applications, multi-user applications require support for more touch points, e.g., to allow for the concurrent modeling of business processes on a multi-touch table.

C5 (Number of Concurrent Users): Multi-touch applications may be concurrently used by multiple users. Obviously, the degree of concurrency is limited by the screen size. While concurrent interactions with a tablet would affect each other, the concurrent use of an application running on a multi-touch table (e.g., joint editing or modeling) might significantly improve the way of working [23]. Especially, this should be exploited in the context of collaborative business process modeling involving multiple process designers [24].

C6 (Usage of Common Interaction Concepts): As known from conventional desktop applications, there are interaction concepts representing de-facto standards. Examples include the menu bar on the top of the application window or *File* as first item of such a menu bar. Such recurrent patterns help users to intuitively interact with applications. The same applies to multi-touch applications and gesture-based interactions respectively; e.g., the *pinch gesture* is typically used for zooming. However, there are only few gestures that have been used in a consistent manner so far. Therefore, [25] suggests a gesture dictionary with the goal that different applications show similar behavior in the context of a specific gesture.

C7 (Intuitive Gestures): Gestures used in the context of multi-touch application should be as simple and intuitive as possible. This supports the user in memorizing und applying them. Regarding a simple gesture, studies have shown that 8 seconds are required to plan and perform it. In turn, complex gestures require 15 seconds [18]. Generally, the simpler a gesture is, the more intuitive its use will be [18].

The introduced characteristics indicate the wide range of possible applications running on multi-touch devices. Obviously, these characteristics should be also taken into account when designing a gesture set for business process modeling.

Required Functions for Modeling Business Processes

This sub-section introduces a core function set required to define and change business process models. This function set is derived from the change patterns originally presented in [26], e.g., to add, delete, move, or replace process activities and process fragments respectively. Empirical evi-

dence of these patterns is provided in [27], while [28] describes their formal semantics.

In the context of this paper, we further assume that process models are *block-structured* and based on an activity-centric process modeling language; we refer to [29] for a discussion of the advantages offered by block-structured process models. Furthermore, process models using advanced algorithms [30], i.e., the process designer cannot position single process elements arbitrarily on the screen. Finally, an initial process model consists of a start and an end node connected through a control-flow edge.

Based on the process modeling functions F1-F8, which are introduced in the following, block-structured process models can be created ensuring that their elements are connected and properly nested [31].

Function *F1 (Insert Activity)* inserts a single activity into a process model between two existing nodes, while function *F2 (Insert Surrounding Block)* encloses an existing process fragment, having single start and end nodes with an XOR (AND) block, i.e., an XOR (AND) split with a corresponding XOR (AND) join gateway. The enclosed process fragment may be also "empty" leading to the creation of an XOR (AND) block with one "empty" branch. Finally, function *F3 (Insert Branch)* extends an existing XOR (AND) block with an empty branch between the splitting and joining gateway. Function *F4 (Renaming Element)* changes the name of a process element, e.g., an activity or data element. To delete process elements, function *F5 (Delete Element)* can be applied. In our context, the process element to be deleted may be an activity, a data element, or a gateway. Comparable to function F1, function *F6 (Insert Data Element)* adds a new data element to the process model. In turn, the added data element can be connected to activities using function *F7 (Insert Data Edge)*. Finally, function *F8 (Extract/Inline Sub-process)* either extracts a sub-process based on a selected set of activities or it inlines this sub-process. In particular, this function is useful in the context of process model refactorings [32, 33], and is especially required when modeling complex process hierarchies.

An overview of this core function set is given in Table 1.

Core Function Set	
F1 Insert Activity	F5 Delete Element
F2 Insert Surrounding Block	F6 Insert Data Element
F3 Insert Branch	F7 Insert Data Edge
F4 Renaming Element	F8 Extract/Inline Sub-process

Table 1. Core Function Set for Changing Process Models

In this paper, we restrict ourselves to this core function set, which provides elementary functions required to create and change process models. Please note that moving or replacing of an activity may be expressed through delete and insert functions. Obviously, additional functions are required for a sophisticated process modeling tool (e.g., to update process attributes or to undo/redo modeling steps).

AN EXPERIMENT ON MULTI-TOUCH PROCESS MODELING

In the context of business process management, experimental research has already shown promising results regarding user-centric process support [34–37]. In our research towards gesture-based process modeling, we conducted a user experiment related to multi-touch business process modeling. Based on the results of this experiment, we will develop a core gesture set for realizing modeling functions F1-F8.

Using the *Goal Definition Template* presented in [38], the goal of our experiment can be defined as follows:

Analyze	*multi-touch process modeling interaction*
for the purpose of	*evaluating*
with respect to their	*intuitive usage*
from the point of view of	*the researchers and developers*
in the context of	*students and research assistants.*

Based on this goal definition, our experiment evaluates multi-touch interactions applied by users or – to be more precise – by students and research assistants when defining and changing process models on multi-touch devices. Experiment results are then used to develop a core set of modeling gestures covering process modeling functions F1-F8.

Based on the above goal definition, following research question is derived:

Do users intuitively use multi-touch gestures when modeling and changing process models on multi-touch devices?

The experimental setup described in the following refines this goal and provides details about the experiment. The subsequent section then presents the design of the experiment, while the last section discusses experiment results.

Experimental Setup

This section describes the subjects, object, and response variable of our experiment.

Subjects. In companies, usually, business process models are documented by dedicated process designers [31]. While, in some cases these designers only obtain a basic training, in others they have been familiar with business process modeling for a long time. Accordingly, from our subjects we require at least a basic knowledge in process modeling (i.e., they are able to apply the process modeling language correctly).

Object. The object of the experiment is a process model expressed in terms of *BPMN, i.e., Business Process Model and Notation* [39]. To be more precisely, this process model deals with the process of shipping a package (cf. Figure 1).

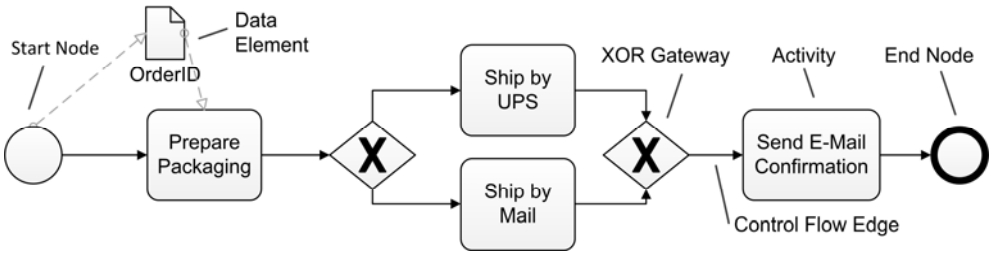


Figure 1. Initial Business Process Model

Each subject shall change this process model in eight different modeling steps using multi-touch gestures (cf. Table 2). Thereby, each step includes a description explaining to the subject what shall be done in order to perform the respective modeling step. For example, modeling step S1 (cf. Table 2) can be accomplished using core function *F1 (Insert Activity)* as introduced in the background section. In particular, we are interested in understanding which gestures users prefer to accomplish the different modeling steps.

S1	Think of a way to add an activity between activities “Prepare Packaging” and the XOR branch.
S2	Label the newly added activity as “Print Invoice”. How would you realize this labeling with a gesture?
S3	Find a way to connect data element “OrderID” with the newly added activity “Print Invoice” using a read data edge.
S4	Insert an empty branch between the two XOR gateways.
S5	Insert an AND block that surrounds the XOR block and activity “Send E-Mail Confirmation”. Note that the resulting process model shall be block-structured.

S6	Add a data element labeled “Package Number”. This data element shall be written by activity “Ship by UPS” and be read by activity “Send E-Mail Confirmation”.
S7	Combine the XOR block and activity “Send E-Mail Confirmation” to a sub-process
S8	Delete the previously added activity “Print Invoice”.

Table 2. Modeling Steps of the Experiment

Factor and factor levels. The *factor* considered by our experiment is *modeling experience* of each subject; factor level is either *beginner* or *expert*.

Response variable. The response variable of our experiment is the *interaction category* used to perform the requested change of the process model. Therefore, for each modeling step,the response variable may have one of the following values: *gesture-based*, *picture-based*, or *menu-based interaction*. The first category groups *gesture-based interactions*. This kind of interaction uses simple movements of fingers on the multi-touch screen when changing the process model. This interaction corresponds to physical gestures as described in the background section. The second category comprises *picture-based interactions*, i.e., all interactions drawing a realistic representation of the final result expected from the application of the respective modeling function. This interaction corresponds to symbolic gestures as described in the background section. Finally, the third category groups *menu-based interactions*, i.e., process model elements are picked from a menu bar or context bar to change the process model. This interaction category is typically used to apply a modeling function within a desktop application.

Experimental Design

We apply the guidelines for the design of experiments as described in [38] and conduct a *multi-test within object study*, i.e., having two groups of subjects and one object. *Students* of Computer Science (as first group) do not have a broad background in process modeling and process management, whereas the second group consists of *research assistants* being experienced with process modeling over years (i.e., researchers employed at our institute at Ulm University).

Instrumentation and data collection procedure. As instrumentation of the experiment, an Apple iPad and the multi-touch drawing application *Doodle Buddy* [40] are used. For each modeling step, this application displays an image to the subject. This image depicts a process model serving as starting point for the model change to be conducted in the respective step. Furthermore, the aforementioned textual explanation is displayed (cf. Table 2). Finger movements of subjects are captured through lines overlaying the im-

age (cf. Figure 3). Additionally, the gesture is captured through a video camera. The latter records the screen of the device as well as the hands of the subject and their movements. Subjects are asked to *think aloud* about what they are doing and what they are thinking during process modeling [41]. This information is used for classifying and interpreting results afterwards. Furthermore, the supervisor of the experiment stays in the same room as the subject, motivates him or her to *think aloud*, and provides assistance in case of emerging questions. Finally, demographic data and qualitative feedback is gathered from subjects based on paper-based questionnaires. Figure 2 gives an overview of the instrumentation used.

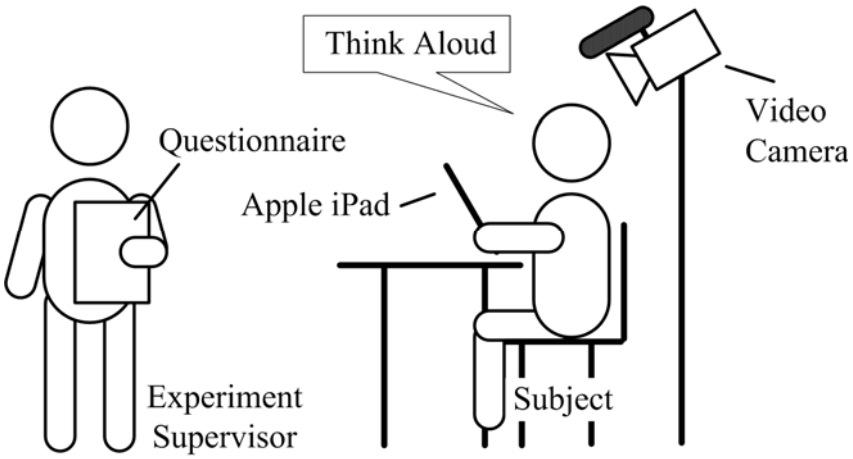


Figure 2. Experimental Design

Experiment Analysis and Results

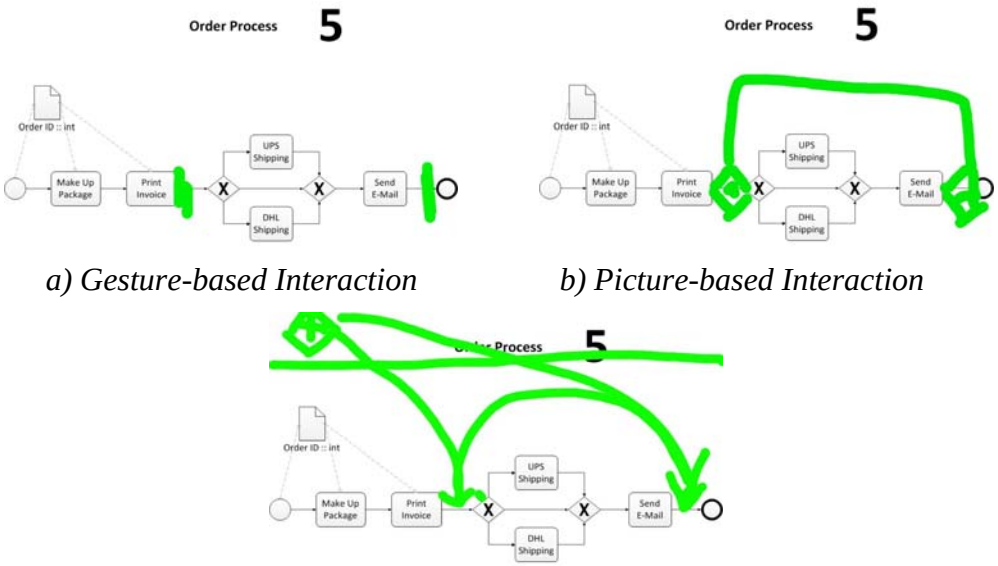
26 subjects attended the experiment. Table 3 summarizes background information about them. It is noteworthy that the number of subjects experienced with multi-touch devices (e.g., smartphones) and the number of subjects not using multi-touch devices is almost equal.

Total	**Gender**	**Profession**	**Handedness**	**Multi-Touch Experience**
26 Subjects	19 Male 7 Female	17 Students 9 Res.Assists.	25 Right 1 Left	14 Yes 12 No

Table 3. Subjects‘ Background

As already described, we classify the interactions a subject applies in the context of a concrete modeling step into three *interaction categories, i.e., picture-based, gesture-based,* and *menu-based interaction*. Note that this classification is accomplished by two process modeling experts and there-

fore constitutes a subjective measurement. As basis of this measurement, we choose the drawing that overlays the image corresponding to the respective modeling step as well as the video capturing the actual movement of the subject's hand. Figure 3 shows results for each of these interaction categories. The subject corresponding to Figure 3a applies a *gesture-based interaction* to insert a surrounding AND block by moving two fingers up and down at the respective insert positions. In Figure 3b, the subject draws a symbolic representation of the surrounding AND block, which refers to a *picture-based interaction*. Finally, in Figure 3c, a subject draws a menu-bar at the top and drags & drops the AND gateways to the respective position in the process model (i.e., *menu-based interaction*).



a) Gesture-based Interaction *b) Picture-based Interaction*

c) Menu-based Interaction

Figure 3. Interaction Categories of the Experiment

Accordingly, Figure 4 summarizes the results of the experiment and visualizes the distribution of the interaction categories. On the x-axis, each step of the experiment is represented through the corresponding function needed to accomplish this step. In turn, the y-axis shows the number of subjects using interactions from a specific category.

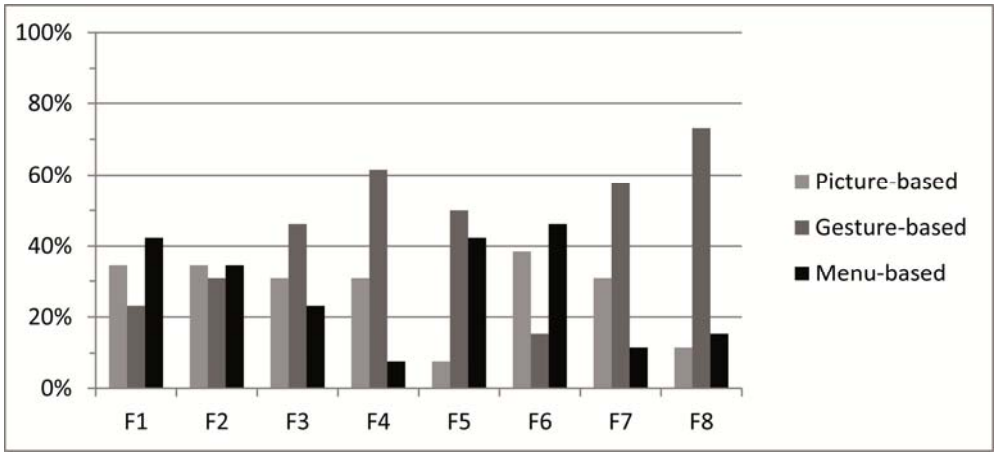


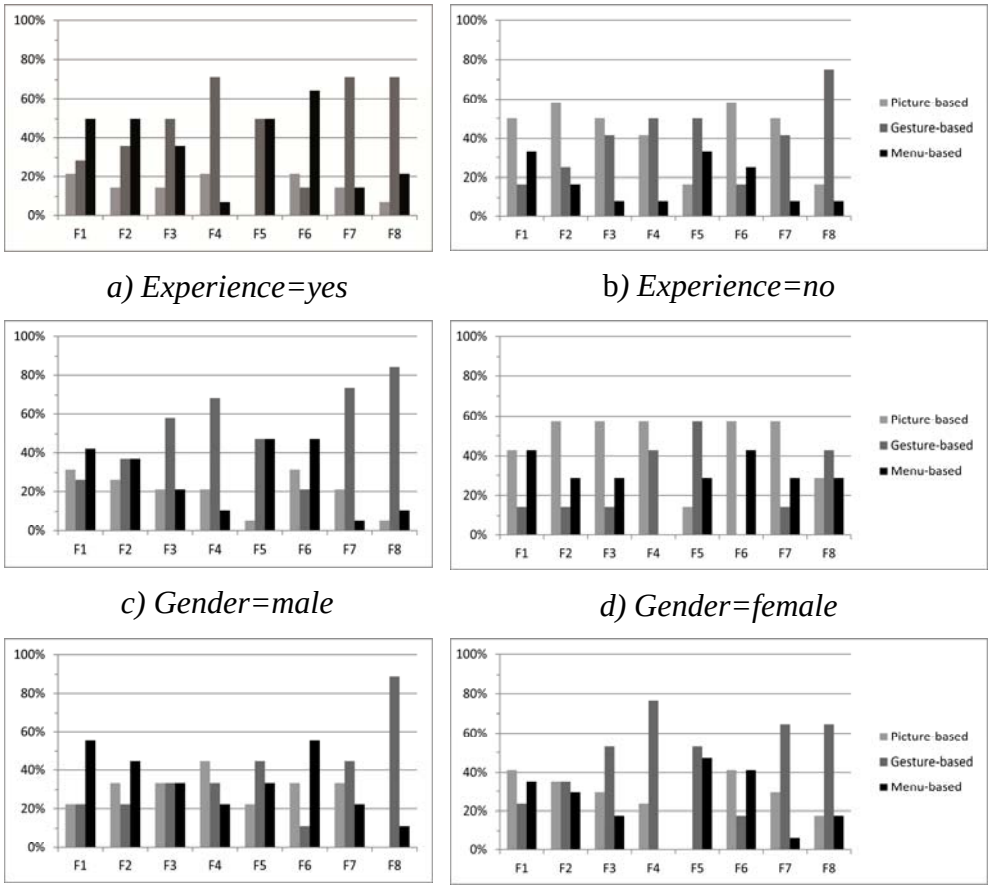
Figure 4. Distribution of Interaction Categories

Obviously, there is no predominant interaction concept covering all process modeling functions needed. However, for certain functions one can observe a clear preference for a specific interaction concept. For example, function *F8 (Extract/Inline Sub-process)*, is applied by 73% of the subjects using gesture-based interaction. However, for other functions (e.g., *F2 (Insert Surrounding Block*)) no dominating interaction concept can be identified. Hence, we may conclude that multiple interaction concepts are required to optimally support different user groups in applying respective process modeling functions. Comparing this result with our research question, there is a tendency that users prefer gesture-based interactions, i.e., for five of eight functions most users apply a gesture-based interaction.

When drilling down results, differences of the interactions applied by the two groups of subjects can be identified. Analyzing the influence of multi-touch experience, for example, shows that experienced subjects use gesture-based interactions more often than picture- and menu-based interactions (cf. Figure 5a). By contrast, unexperienced subjects tend to predominantly use picture-based interactions, but do hardly use menus (cf. Figure 5b). Probably, this can be explained by the fact that unexperienced subjects are unaware of common interaction concepts as provided by multi-touch devices (see the discussion provided in the background section). In the context of our research questions, this may indicate that the statement is only true for users with multi-touch experience.

Comparing the results of the two gender groups (cf. Figure 5c+d), female subjects predominantly use picture-based interactions. Moreover, none of the female subjects uses gesture-based interactions when applying function *F6 (Insert Data Element)*. In turn, male subjects show a clear preference for gesture-based interactions and the research question may be true for them.

Finally, comparing the results of the professions, subjects having lower experience with process modeling (i.e., students) slightly prefer gesture-based interactions (cf. Figure 5f). Opposed to this, subjects having a more profound modeling experience (i.e., research assistants) do not clearly prefer a specific interaction except for function F8 (cf. Figure 5e). Overall, no clear distinction can be made in this context. Our research question may be proven for both professions.



a) Experience=yes b*) Experience=no*

c) Gender=male *d) Gender=female*

e) Profession=Res.Assist. *f) Profession=Student*

Figure 5. Experiment Results

Note that the results of our experiment are only representative to a certain degree due to the rather low number of subjects and the chosen experiment setting. Nevertheless, when taking the insights we gained from the recorded *videos* and *think aloud sessions* into account as well, it can be concluded

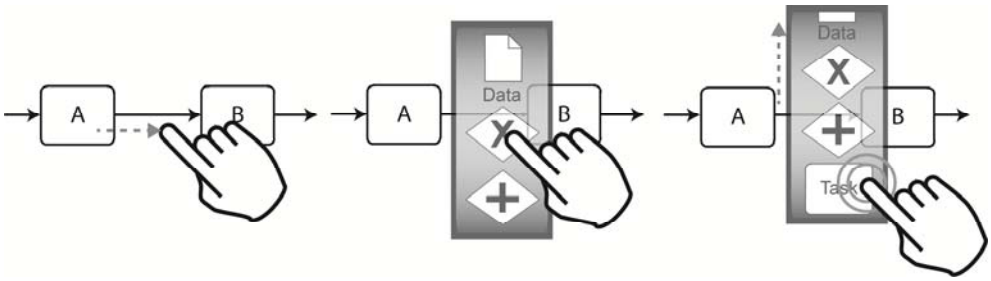
that process modeling experience does not influence the way how processes are modeled on multi-touch devices. By contrast, being experienced with multi-touch devices affects the way of modeling processes on them. This can be explained by the non-availability of standardized gestures on multi-touch devices as already discussed in the background section. Therefore, when developing a multi-touch application, its target group should be taken into account as well; e.g., it might be relevant to know whether members of the target group use their own devices or multi-touch devices are handed out to them.

In the following, we focus on end users experienced with multi-touch devices, we propose a core gesture set enabling intuitive process modeling.

A CORE GESTURE SET FOR PROCESS MODELING

Taking the results of our experiment into account, we propose a core gesture set for creating and changing process models. In this context, we must cope with the trade-off existing between common interaction concepts as provided by multi-touch devices and the results of our experiment. Our goal is to realize an appropriate gesture set suited for defining and changing process models on multi-touch devices.

Our experiment has shown that users prefer a menu-based interaction when adding new elements to a process model (cf. Figure 4). Therefore, our core gesture set contains gesture *G1 (Swipe/Scroll)*, which includes a slider menu enabling the insertion of activities, data elements, and surrounding control-flow blocks (cf. functions F1, F2, F6). To trigger the slider menu, users wipe with their finger from an existing process element to the right in order to insert the new element at the desired position (cf. Figure 6a). Following this, the slider menu appears at the position where the user releases his or her finger from the surface of the multi-touch device.



a) Swipe to Right *b) Scroll the Menu* *c) Tap on Element*

Figure 6. Gesture for Inserting New Elements

Through up and down movements in the slider menu, the required process element can be chosen. For example, to insert an element the user taps on

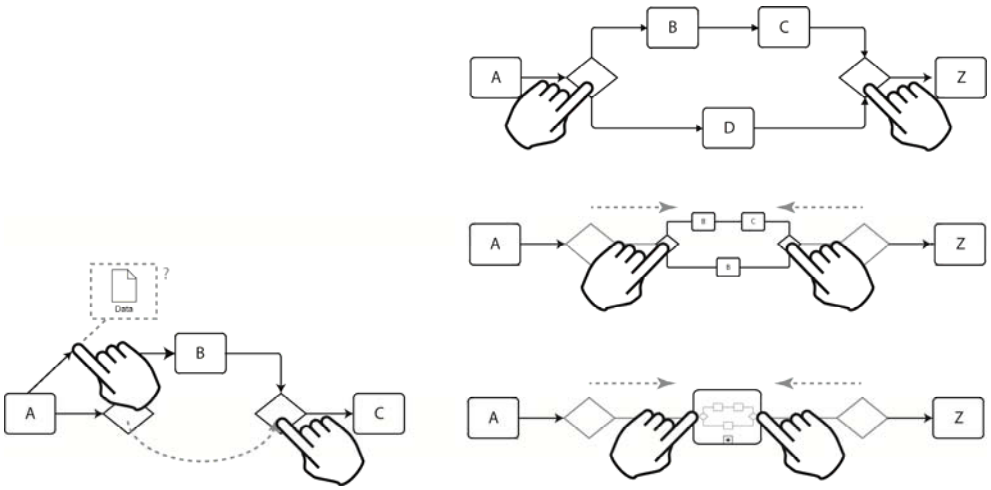
the respective icon in the slider menu (cf. Figure 6c); afterwards the slider menu disappears and the element is inserted.

Obviously, when inserting a process fragment, a second position needs to be selected in order to add the corresponding join gateway. For this purpose, the process model editor shows valid positions in the process model where the joining gateway may be inserted; the user then chooses a position by tapping on it. Note that a block-structured process model allows for such a user assistance [30, 42].

To provide even more sophisticated end user support abstractions in terms of *process views* are useful; e.g., a view abstracts the original process model to a simpler one only comprising those activities relevant for the insertion of a join gateway in a given context [11, 12].

The ordering of the process elements depicted in the slider menu (cf. Figure 6) can be additionally optimized considering their usage frequency. For example, inserting an activity might be more often required compared to the insertion of a surrounding control-flow block. Therefore, the respective function should be positioned on a "central" position in the slider menu such that it can be quickly accessed.

Core Gesture *G2 (Draw Edge)* allows inserting new edges into a process model (cf. Figure 7a). Since we presume a block-structured modeling approach with a fixed layout [30], in general, only two types of edges need to be "manually" added: data edges on one hand and edges connecting split and join gateways in order to add "empty" branches on the other. For example, the user must draw a line with his finger between a split and a corresponding join gateway if he wants to insert an "empty" branch between them (i.e., function F3).



a) Insert Edges *b) Extract/Inline Sub-processes*

Figure 7. Gestures to Insert Edges and Extract/Inline Sub-processes

In turn, to insert a data edge between an activity and a data element read or written by it (i.e., function F7), the user has to draw a line using core gesture *G2 (Draw Edge)*. Thereby, the direction of the respective data edge indicates whether it represents a read or write access of the activity on the data element. Furthermore, when inserting a data edge, the gesture set supports users by suggesting a target element (see the left hand side of Figure 7a). If the target element is the desired one, the user can lift his finger and the data edge will be automatically completed and inserted. Note that this gesture is in accordance with our experiment results since for both functions (i.e., F3 and F7) most subjects have preferred a gesture-based interaction.

Taking our experiment results into account, we designed core gesture *G3 (Pinch Control Flow Block)* to extract or inline a sub-process (i.e., function F8). Regarding this two-handed gesture, the user pushes both the start and end element of the process fragment to be extracted from the process model and replaced by a complex activity (i.e., sub-process) (cf. Figure 7b). While pinching the two elements towards each other, the movement needs to be animated to give direct feedback to the user. When completing the gesture, the resulting sub-process model is linked to a complex activity. In turn, the sub-process model can then be displayed by applying a double tap gesture on this complex activity. To inline a sub-process, the user pinches the sides of the complex activity, referring to this sub-process, apart. Then, the sub-process model appears and is inlined into the overall process model, i.e., it then replaces the node of the complex activity. Note that this gesture matches with the results of our experiment, for which a majority of 73% of the subjects use gesture-based interaction to realize function F8.

Though 62% of the subjects of our experiment have used gesture-based interaction to rename a process element (i.e., free-hand writing on the screen), most mobile applications provide an on-screen keyboard for inserting or modifying text on multi-touch devices. The keyboard is hidden most of the time to save screen space, i.e., it will be only displayed when the user wants to add or modify text. To comply with common interaction standards on multi-touch devices, in our gesture set, function *F4 (Renaming Element)* can be activated through core gesture *G4 (Tap Element)*. In turn, this gesture is triggered by tapping with the finger on a process element. Afterwards the keyboard appears and the user may modify the text. After confirming the modified text, the keyboard disappears.

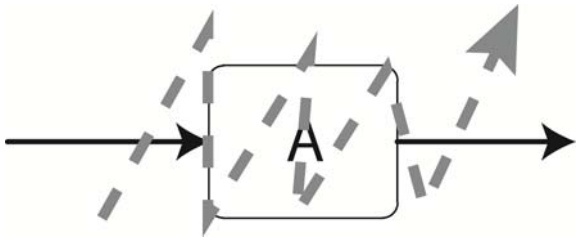


Figure 8. Gesture to Delete Process Element

The majority of subjects have deleted process elements using gesture-based interactions. For realizing this function, we designed core gesture *G5 (Zig-Zag Line)*, which crosses out the element to be deleted (cf. Figure 8). Generally, different variants for realizing this gesture could be envisioned, e.g., drawing two crossing lines or one line from the bottom-left to the top-right. Regarding our core gesture set, the user draws a connected zigzag-line on the respective element. Generally, this is more accurate regarding recognition when compared to the drawing of two separate lines.

Table 4 summarizes the designed core gestures and their mapping on respective modeling functions. The core gestures are kept as simple as possible to allow for their intuitive use (cf. characteristic C7). Finally, we reduce the total number of gestures in order to minimize the mental efforts users have when memorizing the gesture.

Core Modeling Functions	**Multi-Touch Core Gestures**
F1 Insert Activity	G1 Swipe/Scroll
F2 Insert Surrounding Block	G1 Swipe/Scroll
F3 Insert Branch	G2 Draw Edge
F4 Renaming Element	G4 Tap Element
F5 Delete Element	G5 Zig-Zag Line
F6 Insert Data Element	G1 Swipe/Scroll
F7 Insert Data Edge	G2 Draw Edge
F8 Extract/Inline Sub-process	G3 Pinch Control Flow Block

Table 4. Mapping Core Functions to Multi-Touch Gestures

EMBEDDING GESTURE-BASED PROCESS MODELING IN A PROCESS MODELING FRAMEWORK

The core gesture set presented in this paper has been implemented in the context of our *proView* framework; *proView* aims at supporting users in intuitively interacting with large business process models and evolving them. For this purpose, personalized and updatable *process views* (cf. Figure 9,

part 2) can be created for each user abstracting from the overall process model stored in the central process repository (cf. Figure 9, part 1).

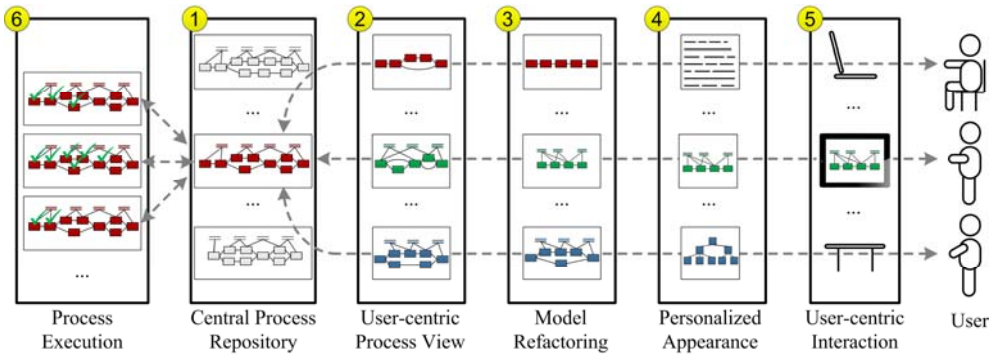


Figure 9. Overview proView Framework

More precisely, such a process view abstracts from the overall business process model by hiding non-relevant process elements (i.e., *reduction operation*) or by combing them (i.e., *aggregation operation*) to an abstracted node. Figure 10 gives an example of how such a process view is constructed in *proView* [12, 43, 44].

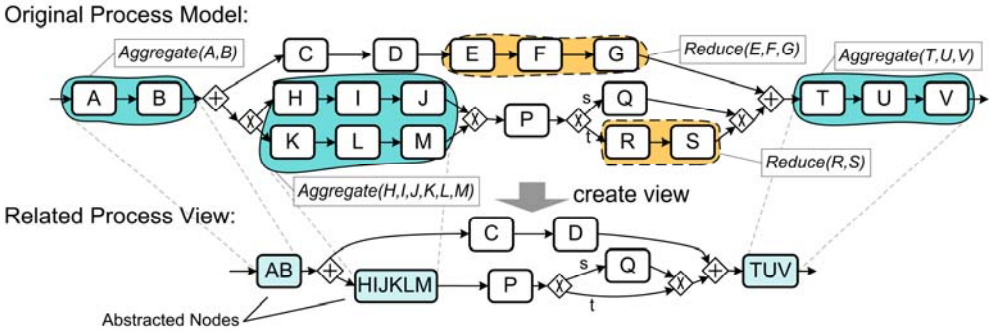


Figure 10. Example of a Process View

When applying view creation operations, non-relevant gateways or empty AND branches might result. Therefore, the model of the process view is further simplified through behavior-preserving refactorings (cf. Figure 9, part 3), e.g., AND gateways of a parallel branching with only one remaining branch may be removed. Furthermore, part 4 of the *proView* framework transforms the process view into a personalized appearance, e.g. a textual, form-based, or tree-based representation. Finally, the result is presented to the user (cf. Figure 9, part 5). In this context, the core gesture set introduced in this paper is embedded in the *proView* framework. In particular, this core gesture set is required since *proView* users may not only use desktop computers, but also multi-touch devices such as smartphones, tablets, and touch tables to visualize and change their business processes. Generally, for hu-

man-centric business process management users should be able to intuitively interact with multi-touch devices, but also to cope with their limitations (e.g., limited screen size). Finally, the *proView* framework supports process execution in process-aware information systems (cf. Figure 9, part 6). In this context, run-time information is added to visualized business process models and process views respectively.

PROOF-OF-CONCEPT IMPLEMENTATION

The implementation of the *proView* framework consists of two major components: *proViewServer* and *proViewMobile* (cf. Figure 11). An instance of the *proViewMobile* client is created for each user. It handles his multi-touch interactions with as well as the visualization of his process models and process views respectively. Furthermore, *proViewMobile* is implemented on an Apple iPad and communicates with the proViewServer using a RESTful protocol.

In turn, the *proViewServer* implements the logic of the *proView* framework and provides engines for process model visualization, change, and execution & monitoring [45, 46]. More precisely, proViewServer represents a *business process* in terms of a *Central Process Model (CPM)*; additionally, so-called *creation sets (CS)* are defined, whereby each CS specifies the schema and appearance of one particular process view [8].

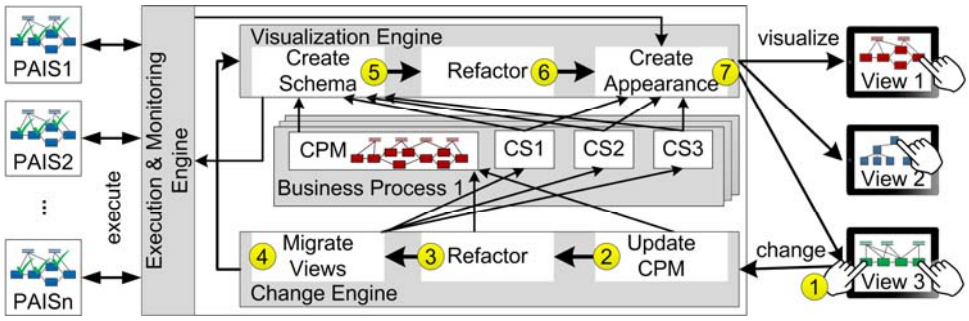


Figure 11. proView Architecture

When a user updates a process view by applying multi-touch gestures in *proViewMobile*, the *change engine* of the *proViewServer* is triggered (see Step 1 in Figure 11). First, the view-based model change is propagated to the CPM using well-defined propagation algorithms (see Step 2 in Figure 11). Next, the CPM is simplified (see Step 3 in Figure 11), i.e., behaviour-preserving refactorings [32] are applied to foster model comprehensibility (e.g., by removing gateways not needed anymore since the respective branching exactly comprises one branch after the change). Afterwards, the creation sets of all views associated with the CPM are migrated to the new

CPM version (see Step 4 in Figure 11). This becomes necessary since a creation set may be contradicting with the changed CPM. Finally, all views are recreated.

Generally, in *proView*, the *visualization engine* generates a process view based on a given CPM and creation set CS, characterizing the view-specific adaptions, i.e., the CPM is transformed into the view model by applying the *view creation operations* specified in CS (see Step 5 in Figure 11). Afterwards, the obtained view is *simplified* by applying well-defined *refactoring operations* (see Step 6 in Figure 11). Step 7 then customizes the visual appearance of the view; e.g., by creating a tree-, form-, or flow-based visualization. Finally, the updated process view is sent to the *proViewMobile* client and displayed by it.

Figure 12 shows a screenshot of proViewMobile. The user interface is kept rather simple, and implements a *hybrid interaction* (as described in the background section). The user interface has a tab bar on the bottom of the screen, which enables the user to open the *process list* and *configuration window*. The *process list* then displays all CPMs and associated process views available on the proViewServer. By clicking on a list item, the respective process model is loaded from the server. In turn, in the *configuration window* the address of the proViewServer as well as login information are stored. The remaining screen is used to display the selected process model.

To scroll on the screen, a two finger gesture is used, i.e., the user taps down with two fingers and moves around the screen to scroll.

For example, consider Figure 13 depicting how core gesture *G1 (Swipe/ Scroll)* for inserting process elements has been realized. First, the user swipes to the right (or left). Meanwhile, a progress bar on the top of the process element indicates that the gesture has been recognized and its application is progressing (cf. Figure 13a). When the progress bar is filled and the finger is released, a context menu opens. If the bar is not filled and the finger is released, the gesture will be dismissed. The progress bar is required to give the user direct feedback of what he is doing. Once opened, the context menu offers the possibility to hide, delete or insert a process element (cf. Figure 13b). By clicking on an item the respective modeling function is performed (cf. background section).

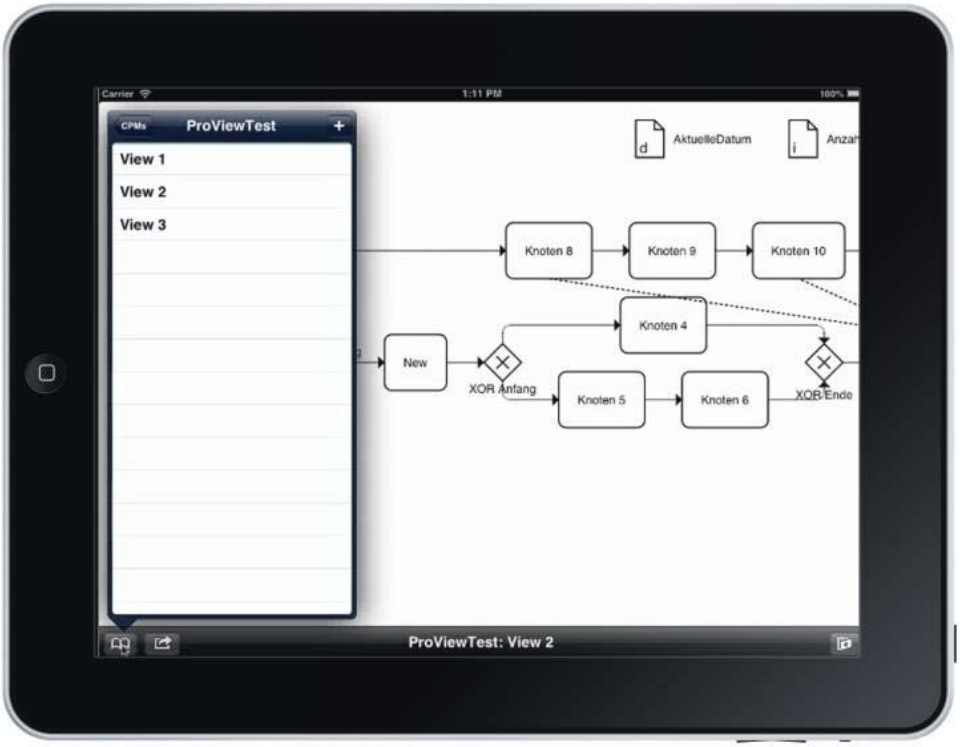
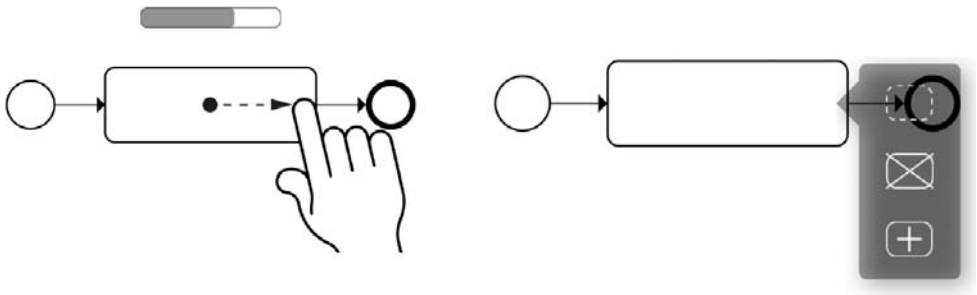


Figure 12. Screenshot of proViewMobile

Note that core gesture *G5 (Zig-Zag Line)*, we designed for deleting process elements, has not been fully implemented in that way so far, since no native gesture recognizers supports this gesture in Apple iOS.

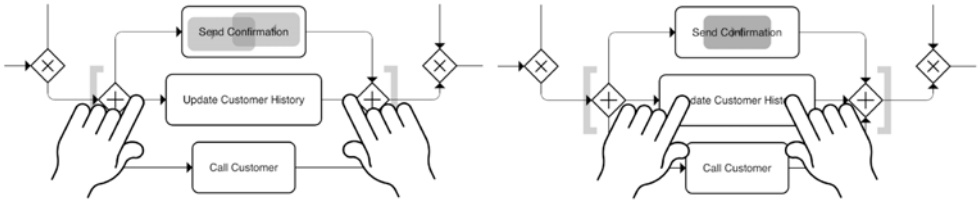


a) Swipe Gesture to Open Menu *b) Resulting Menu*

Figure 13. Implemented Gesture to Hide, Delete, and Insert

Figure 14 shows the implementation of core gesture *G3 (Pinch Control Flow Block)* based on which a sub-process can be extracted or inlined. When applying this gesture, the user touches the start and end point of the process fragment to be aggregated and starts moving his fingers towards each other (i.e., pinch gesture). To give visual feedback to the user, at the

start and end point proViewMobile draws squared brackets (cf. Figure 14a). While moving with fingers, two grey boxes are moving towards each other. At the point they overlap completely, the fingers can be released to complete the gesture. Otherwise, the gesture is dismissed.



a) Begin of Pinch Gesture *b) Function Ready to Execute*

Figure 14. Implemented Gesture to Extract/Inline Sub-process

Core gesture *G4 (Tap Element)* renames elements in a process model. The proViewMobile client implements gesture G4 in terms of a long tap, i.e., the users touches the screen with one finger for two seconds on the respective process element in order to open the keyboard for naming this element. The long tap is applied to avoid non-intended usage of the gesture.

Using the proViewMobile client, further experiments will be conducted to evaluate the usability of the developed core gesture set and to elaborate emerging opportunities for modeling processes using multi-touch devices.

RELATED WORK

The background section has already given profound insights into multi-touch interaction concepts and characteristics of multi-touch applications. This section discusses concrete approaches targeting at user support and user interaction in the context of human-centric business process management.

Assisting end users in understanding and changing large as well as complex process models is crucial in order to successfully implement business process management in agile companies [6, 47]. In this context, several experiments has been conducted to understand how users, having hardly any technical background, model business processes [36, 48, 49]. Although these works provide important insights into the processes of process modeling and changing processes, none of them addresses multi-touch process modeling.

Various approaches for intuitively creating process models through tangible objects with and without multi-touch tables exist. Hence, each process element is represented through a physical object [24, 50]. Since the screen size

of such devices is limited, these approaches are unable to cope with large business process models.

Other approaches use virtual environments to visualize processes and to coach users in respect to their role in the context of a particular process. For example, such a virtual environment may consists of a 3D hospital and may visualize a patient treatment process [51]. Respective techniques might contribute to a better understanding of business processes. However, manually setting up such virtual environments is time-consuming and therefore not suitable for most companies. Furthermore, changing business process models through users is not addressed at all in such virtual solutions.

Virtual environments are also used to collaboratively model business processes. Thereby, process designers as well as process models correspond to 3D objects in the virtual environment [52]. Virtual environments are also used to visualize process models as well as their simulation data [53]. Both approaches do not focus on users having limited technical background. Furthermore, they do not take multi-touch interaction into account.

Sonification is used to make process models better accessible to users, e.g., through visible as well as audible information [54]. This promising research is at a rather early stage and has not yet considered abstraction techniques to cope with large process models comprising hundreds of activities.

Our *proView* framework provides a holistic framework for personalized process views and focuses on the interaction with the end-user [8]. Further, it enables users to change business processes based on their views through multi-touch gestures, and it allows this change to the overall process model. Existing approaches neither cover these aspects in an integrated way nor are they based on rigid constraints nor do they take practical requirements into account.

SUMMARY AND OUTLOOK

This paper has presented the results of an experiment we performed to investigate how users define and change process models on multi-touch devices in a natural and intuitive way. Experiment results indicate that gestures are useful for realizing broadly required process modeling functions; however, it also has become evident that for some functions menus are preferred when editing or changing a process model.

Based on the experiment results achieved as well as common interaction concepts of multi-touch applications, a core gesture set has been designed. This gesture set provides the basis for process modeling tools offering new ways of gesture-based process modeling, while ensuring a high degree of usability. Note that it would be no good idea to simply migrate existing modeling tools straightforward to multi-touch devices, since these tools

were primarily designed for desktop computers and hence usability would be affected. The gesture set we developed is independent from a concrete device type (e.g., table, tablet or smartphone), but has to be extended with gestures for supportive functions (e.g., undo/redo, copy, paste) in order to be able to broadly applicable in a production environment. Furthermore, the core gesture set is implemented as native Apple iPad application communicating with the proView framework.

As next step, our proof-of-concept prototype will be extended in order to support multi-touch tables as well. Using this modeling application, further experiments will be conducted to evaluate cooperative process modeling on multi-touch devices.

References

1. Pryss, R., Langer, D., Reichert, M., Hallerbach, A.: Mobile Task Management for Medical Ward Rounds - The MEDo Approach. Proc BPM'12 Workshops, 1st Int'l Workshop on Adaptive Case Management, 2012, pp. 43-54.
2. Pryss, R., Tiedeken, J., Kreher, U., Reichert, M.: Towards Flexible Process Support on Mobile Devices. Proc CAiSE'10 Forum, 2010, pp. 150–165.
3. Frisch, M., Heydekorn, J., Dachselt, R.: Diagram Editing on Interactive Displays Using Multi-Touch and Pen Gestures. Proc 6th Int'l Conf on Diagrammatic Representation and Inference, 2010, pp. 182–196.
4. Gong, J., Tarasewich, P.: Guidelines for Handheld Mobile Device Interface Design. Proc DSI Annual Meeting, 2004, pp. 3751–3756.
5. Wang, X., Ghanam, Y., Maurer, F.: From Desktop to Tabletop: Migrating the User Interface of AgilePlanner. Proc 2nd Conf on Human-Centred Software Engineering (HCSE'08), 2008, pp. 263–270.
6. Kabicher-Fuchs, S., Rinderle-Ma, S., Recker, J., Indulska, M., Charoy, F., Christiaanse, R., Dunkl, R., Grambow, G., Kolb, J., Leopold, H., Mendling, J.: Human-Centric Process-Aware Information Systems (HC-PAIS). CoRR abs/1211.4, 2012.
7. Rudner, B.: Fortgeschrittene Konzepte der Prozessmodellierung durch den Einsatz von Multi-Touch-Gesten, Bachelor thesis, University of Ulm, 2011.
8. Kolb, J., Kammerer, K., Reichert, M.: Updatable Process Views for User-centered Adaption of Large Process Models. Proc 10th Int'l Conf on Service Oriented Computing (ICSOC'12), 2012, pp. 484–498.
9. Kolb, J., Hübner, P., Reichert, M.: Automatically Generating and Updating User Interface Components in Process-Aware Information Systems. Proc 10th Int'l Conf on Cooperative Information Systems (CoopIS 2012), 2012, pp. 444–454.

10. Kolb, J., Hübner, P., Reichert, M.: Model-Driven User Interface Generation and Adaptation in Process-Aware Information Systems. Technical Report, UIB-2012-04, University of Ulm, 2012.
11. Bobrik, R., Bauer, T., Reichert, M.: Proviado - Personalized and Configurable Visualizations of Business Processes. Proc 7th Int'l Conf Electronic Commerce & Web Technology (EC-WEB'06), 2006, pp. 61–71.
12. Reichert, M., Kolb, J., Bobrik, R., Bauer, T.: Enabling Personalized Visualization of Large Business Processes through Parameterizable Views. Proc 26th Symposium on Applied Computing (SAC'12), 2012, pp. 1653–1660.
13. Kolb, J., Reichert, M.: Using Concurrent Task Trees for Stakeholder-centered Modeling and Visualization of Business Processes. Proc S-BPM ONE, 2012, pp. 237–251.
14. Kolb, J., Rudner, B., Reichert, M.: Towards Gesture-based Process Modeling on Multi-Touch Devices. Proc 1st Int'l Workshop on Human-Centric Process-Aware Information Systems (HC-PAIS'12), 2012, pp. 280–293.
15. Brandl, P., Leitner, J., Seifried, T., Haller, M., Doray, B., To, P.: Occlusion-Aware Menu Design for Digital Tabletops. Proc 27th Int'l Conf on Human Factors in Computing Systems (CHI EA'09), 2009.
16. Rossini, N.: Reinterpreting Gesture as Language. Language in Action. IOS Press, US, 2012.
17. Schegloff, E.A.: On Some Gestures' Relation to Talk. In: Atkinson, J.M. (ed.) Structures of Social Action. pp. 266–296. Cambridge University Press, 2010.
18. Wobbrock, J.O., Morris, M.R., Wilson, A.D.: User-Defined Gestures for Surface Computing. Proc 27th Int'l Conf on Human Factors in Computing Systems (CHI'09), 2009, pp. 1083–1092.
19. Benko, H., Wilson, A.D., Baudisch, P.: Precise Selection Techniques for Multi-Touch Screens. Proc SIGCHI Conf on Human Factors in Computing Systems (CHI'06), 2006, pp. 1263–1272.
20. Wang, F., Ren, X.: Empirical Evaluation for Finger Input Properties in Multi-Touch Interaction. Proc 27th Int'l Conf on Human Factors in Computing Systems (CHI'09), 2009, pp. 1063–1072.
21. Yee, W.: Potential Limitations of Multi-touch Gesture Vocabulary: Differentiation, Adoption, Fatigue. Proc 13th Int'l Conf on Human-Computer Interaction (HCI'09), 2009, pp. 291–300.
22. Gemmell, M.: iPad Multi-Touch. http://mattgemmell.com/2010/05/09/ipad-multi-touch/, last visited: 06.02.2012.
23. Hornecker, E., Marshall, P., Dalton, N.S., Rogers, Y.: Collaboration and Interference: Awareness with Mice or Touch Input. Proc 2008 ACM Conf. on Computer Supported Cooperative Work (CSCW'08), 2008, pp. 167–176.

24. Edelman, J., Grosskopf, A., Weske, M., Leifer, L.: Tangible Business Approach Process Modeling: A New Approach. Proc Int'l Conf on Engineering Design (ICED'09), 2009, pp. 489–500.

25. Elias, J.G., Westerman, W.C., Haggerty, M.M.: Multi-Touch Gesture Dictionary. US Patent & Trademark Office, US7840912, 2007.

26. Weber, B., Rinderle, S., Reichert, M.: Change Patterns and Change Support Features in Process-Aware Information Systems. Proc CaiSE'07, 2007, pp. 574–588.

27. Weber, B., Reichert, M., Rinderle-Ma, S.: Change Patterns and Change Support Features - Enhancing Flexibility in Process-Aware Information Systems. Data Knowl Eng, 66: 438–466. 2008.

28. Rinderle, S., Reichert, M., Weber, B.: On the Formal Semantics of Change Patterns in Process-Aware Information Systems. Proc 27th Int'l Conf Conceptual Modeling (ER'08), 2008, pp. 279–293.

29. Li, C., Reichert, M., Wombacher, A.: Mining Business Process Variants - Challenges, Scenarios, Algorithms. Data Knowl Eng, 70:409–434, 2011.

30. Rinderle, S., Bobrik, R., Reichert, M., Bauer, T.: Business Process Visualization - Use Cases, Challenges, Solutions. Proc 8th Int'l Conf on Enterprise Information Systems (ICEIS'06), 2006, pp. 204–211.

31. Reichert, M., Weber, B.: Enabling Flexibility in Process-Aware Information Systems - Challenges, Methods, Technologies. Springer (2012).

32. Weber, B., Reichert, M., Mendling, J., Reijers, H.A.: Refactoring Large Process Model Repositories. Computers in Industry, 62:467–486, 2011.

33. Weber, B., Reichert, M.: Refactoring Process Models in Large Process Repositories. Proc CaiSE'08, 2008, pp. 124–139.

34. Weber, B., Mutschler, B., Reichert, M.: Investigating the Effort of Using Business Process Management Technology: Results from a Controlled Experiment. Sci Comput Program, 75: 292–310, 2010.

35. Recker, J., Dreiling, A.: Does It Matter Which Process Modelling Language We Teach or Use? An Experimental Study on Understanding Process Modelling Languages without Formal Education. Proc 18th Australasian Conference on Information Systems, 2007, pp. 356–366.

36. Recker, J., Safrudin, N., Rosemann, M.: How Novices Model Business Processes. Proc 9th Int'l Conf Business Process Management (BPM'10), 2010, pp. 29–44.

37. Mutschler, B., Weber, B., Reichert, M.: Workflow Management versus Case Handling: Results from a Controlled Software Experiment. Proc 23rd Annual ACM Symposium on Applied Computing (SAC'08), Special Track on Coordination Models, Languages and Architectures, 2008, pp. 82–89.

38. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslen, A.: Experimentation in Software Engineering - An Introduction. Kluwer Academic Publishers, 2000.

39. OMG: Business Process Management Notation (BPMN) 2.0, www.bpmn.org.

40. Pinger Inc.: Doodle Buddy. http://itunes.apple.com/de/app/doodle-buddy/id313232441?mt=8. , last visited: 06.02.2012.

41. Jaspers, M.W.M., Steen, T., Bos, C. van den, Geenen, M.: The Think Aloud Method: A Guide to User Interface Design. Int'l J Med Informatics, 73:781–795, 2004.

42. Reichert, M., Dadam, P.: ADEPTflex - Supporting Dynamic Changes of Workflows Without Losing Control. Journal of Intelligent Information Systems, 10:93–129, 1998.

43. Kolb, J., Reichert, M.: Data Flow Abstractions and Adaptations through Updatable Process Views. Proc 27th Symposium on Applied Computing (SAC'13), 2013, pp. 1447–1453.

44. Kolb, J., Reichert, M.: A Flexible Approach for Abstracting and Personalizing Large Business Process Models. ACM Appl Comput Rev, 13:6–17, 2013.

45. Kolb, J., Reichert, M.: Supporting Business and IT through Updatable Process Views: The proView Demonstrator. Demo Track of the 10th Int'l Conf on Service Oriented Computing (ICSOC'12), 2012, pp. 460–464.

46. Kolb, J., Kammerer, K., Reichert, M.: Updatable Process Views for Adapting Large Process Models: The proView Demonstrator. Proc of the Business Process Management 2012 Demonstration Track, 2012.

47. Kabicher-Fuchs, S., Kriglstein, S., Figl, K.: Timeline Visualization for Documenting Process Model Change. Proc 5th Int'l Workshop on Enterprise Modelling and Information Systems Architectures, 2012.

48. Neubauer, M., Oppl, S., Stary, C.: Towards Intuitive Modeling of Business Processes- Prospects for Flow- and Natural-Language Orientation. Task Models and Diagrams for User Interface Design, 2010, pp. 15–27.

49. Pinggera, J., Zugal, S., Weber, B.: Investigating the Process of Process Modeling with Cheetah Experimental Platform. Proc 1st Int'l Workshop on Empirical Research in Process-oriented Information Systems, 2010.

50. Oppl, S., Stary, C.: Tabletop Concept Mapping. Proc 3rd Int'l Conf on Tangible and Embedded Interaction, 2009, pp. 275–282.

51. Brown, R., Eichhorn, D., Herter, J.: Virtual World Process Perspective Visualization. Proc 4th Int'l Conf on Information, Process and Knowledge Management (eKNOW'12), 2012.

52. Poppe, E., Brown, R., Recker, J., Johnson, D.: Improving Remote Collaborative Process Modelling using Embodiment in 3D Virtual Environments. Proc Conf in Research and Practice in Information Technology, 2012.

53. Eichhorn, D., Koschmider, A., Yu, L., Sturzel, P., Oberweis, A., Trunko, R.: 3D Support for Business Process Simulation. Proc 33rd Computer Software and Applications Conference (COMPSAC’09), 2009, pp. 73–80.

54. Hildebrandt, T., Kriglstein, S., Rinderle-Ma, S.: Beyond Visualization - On Using Sonification Methods to Make Business Processes More Accessible to Users. Proc 18th Int’l Conf on Auditory Display (ICAD'12), 2012, pp. 248–249.