Edit Distance Modulo Bisimulation: A Quantitative Measure to Study Evolution of User Models

Himanshu Zade[1,2], Santosh Arvind Adimoolam[1], Sai Gollapudi[1], Anind K. Dey[2], and Venkatesh Choppella[1]

{*himanshu.zade, asarvind.adimoolam, saigollapudi1*}@gmail.com, anind@cs.cmu.edu, venkatesh.choppella@iiit.ac.in

[1]Pascal Lab, SERC, IIIT-Hyderabad, India — [2]Ubicomp Lab, HCII, CMU, Pittsburgh, USA

ABSTRACT
When a user learns to use a new device, her understanding of it evolves. A progressive comparison of the evolving *user models* towards the device *target model*, for analysing learning, involves determining the behavioral proximity between them. To quantify the gap between a user model and a target model, we introduce an *edit distance* metric for measuring their behavioral proximity using a bisimulation-based equivalence relation. We define edit distance to be the minimum number of edges and states with incident edges required to be deleted from *and/or* added to a user model to make it bisimilar to the target model. We propose an algorithm to compute edit distance between two models and employ the heuristic procedure on experimental data for computing edit distance between target and user models. The data is organised into two experiments depending on the device the user interacted with: (a) a simple device resembling a vending machine and (b) a close to real-world vehicle transmission model. The results validate our proposed metric as edit distance converges with progressive user learning, increases for erroneous learning, and remains unchanged indicating no learning.

Author Keywords
Learning; finite state machines; behavioral proximity.

ACM Classification Keywords
H.1.2 User/Machine Systems: Human Factors; F.1.1 Models of Computation: Relation between models; D.2.2 Design Tools and Techniques: State diagrams

INTRODUCTION
The rapid proliferation of technology challenges users to quickly become familiar with any new device that they may encounter. User interface (UI) designers, on the other hand, are challenged to provide UIs that are easy to learn and use. In order to do so, they need to understand how a user gets acquainted with a new device. Insight about how a user learns to use a device helps in designing better UIs, creating new interventions to assist learning, quantifying the speed of learning, all of which improve both the efficiency and effectiveness of the user's interaction and experience with the device.

Mental models (*i.e.* mental representations of a user's understanding) are important in learning how to operate a device [24]. We use the term *user model* to refer to a finite state machine (FSM) representing the user's understanding of a device [20, 33]. In contrast to the user model, we use the term *target model* to refer to the FSM representing the behavior of the device, which the end user is trying to learn [37]. For a user to understand the complete behavior of a device, it is essential that her final user model is bisimilar to (*i.e.* its behavior is indistinguishable from) the target model [26]. The user may attain such a model if her multiple user models progressively approach the target model, while interacting with the device to learn it. Relatively little attention has been paid to how user models evolve as a part of a user's learning process.

FSMs have long been used to represent mental models [20]. In this paper, we build on the past work in FSMs, contribute an edit distance metric for comparing the differences between FSMs, and apply this to represent the process of how a user learns to use a new device, by tracing the evolution of user models towards a target model. We study user learning through a progressive comparison of the user model and target model over time. (Bi)simulation relations allow behavioral comparisons between two models [26]; they hold true for a user model indicating complete understanding of a device, false otherwise. However during learning, user models may be incomplete, erroneous and contradictory [14]. It is therefore important to quantify the amount of learning. This translates to the problem of determining the proximity between the two models. (Bi)simulation relations by themselves do not provide such a metric to measure proximity; they capture the notion of order, and not measure. To quantify the gap between a user model and a target model, we introduce *edit distance* for measuring behavioral proximity between them.

The edit distance metric is based on two propositions: (a) A target model can simulate a user model by deleting some edges from the user model. (b) The *edited* user model can now be made bisimilar to the target model by adding some edges and states with incident edges. We define edit distance as the sum of the minimum number of such deletions and additions, as proposed above. We use a heuristic procedure for manually calculating edit distance between FSMs and employ this procedure on our experimental data.

Our proposed representational technique provides an intensional description of the process involved in learning a new device. As the two experiments discussed later in the paper show, it allows us to examine several questions of interest to the HCI community about the learning process: When are two user models close? Do successive user models converge for all users? *etc.* The key here is not to show a mere convergence of the user models, but to capture and represent the process of their evolution to show that the convergence (or divergence) follows a user's ability to learn the system. Such a representation allows designers to identify the problem areas in a UI by differentiating instances when a user model is improving from instances where it is not.

The paper is laid out as follows. First we provide the conceptual foundations for our edit distance approach. We then describe related literature that highlights the need for a novel approach. Then we report on the exploratory study that helped us formulate our solution. Next, we describe the *edit distance* approach in detail and demonstrate the computation of edit distance between two models. We then present our first experimental study that heuristically validates our approach using a simple machine. Later, we strengthen our findings through another study conducted using a close to real-world vehicle transmission system. Using our algorithm, we calculate the heuristic edit distances of successive user models, and demonstrate that they converge or diverge for progressive or regressive user learning respectively, validating our metric.

CONCEPTUAL FOUNDATIONS

This section describes in brief the concepts that are instrumental for understanding the work presented in the paper.

What is an interactive machine?

Most of the devices that people use are interactive in nature. An interactive machine can be succinctly described as a control system whose user inputs can (at least in part) be administered by a user through a UI. Apart from the UI, an interactive machine also has a set of capabilities that a machine offers to the user. It is essential for a user to get acquainted with the UI in order to learn its functional behavior and use it effectively [40]. In our paper, we represent an interactive machine as an FSM, and refer to it as a *target model*.

Finite state machines

An FSM has internal states, transitions between these states and observable outputs associated with each state.

DEFINITION 0.0.1. *An FSM with outputs is a tuple* $\langle X, q, \delta, I, Y, H\rangle$ *where*

- X *is a set of* states, $q \in X$ *is the* start *state,* I *is the set of* user inputs *and* Y *is the set of* outputs.
- $\delta : X \times I \to 2^X$ *is a* transition *function. We write* $s \xrightarrow{a} s'$ *if* $s' \in \delta(s, a)$.
- $H : X \to Y$ *maps each state in* X *to an output in* Y.

An FSM is *deterministic* if $|\delta(s, a)| \leq 1$ for all $s \in X$ and $a \in I$, else *nondeterministic*. In this paper, we restrict ourselves to deterministic FSMs since we model deterministic devices.

Simulation and bisimulation

A simulation relation is a relation between the states of two different FSMs such that whenever there is a transition in one FSM, there is a corresponding transition in another FSM with respect to the relation. A bisimulation relation is a simulation relation whose inverse is also a simulation. Formally, we define the simulation and bisimulation relation as follows.

DEFINITION 0.0.2 (SIMULATION RELATION). *Let* $L_1 = \langle X_1, q_1, \delta_1, I, Y, H_1\rangle$ *and* $L_2 = \langle X_2, q_2, \delta_1, I, Y, H_2\rangle$ *be two finite state machines. A relation* $R \subseteq X_1 \times X_2$ *is called a simulation* of L_1 by L_2 *if, for each pair* $(s_1, s_2) \in R$,

1. $H_1(s_1) = H_2(s_2)$*, i.e., the output of both states is same.*
2. *Whenever* s_1 *makes a transition, then* s_2 *makes a transition such that the corresponding pair of next states are also in the relation* R*,* i.e., *if* $s_1 \xrightarrow{a} s_1'$*, then there exists a state* $s_2' \in X_2$ *such that* $s_2 \xrightarrow{a} s_2'$ *and* $(s_1', s_2') \in R$.

DEFINITION 0.0.3 (BISIMULATION). *A simulation relation* R *of* L_1 *by* L_2 *is also a bisimulation relation between* L_1 *and* L_2 *if* R^{-1} *is a simulation relation of* L_2 *by* L_1.

Simulation establishes that all behaviors in the simulated machine are reflected in the other (simulating) machine. Bisimulation is a two-way simulation that establishes behavioral equivalence between two machines. The concept of simulation and bisimulation thus allows for behavioral comparison and equivalence respectively [26]. Successful simulation of the user model by the target model implies that the user's learning is coherent with the behavior of the machine. However, this does not indicate whether the user has learnt the behavior of the machine entirely. To ascertain that the user's understanding is complete and coherent, we require a bisimulation relation between the user model and the target model encompassing all the states of both the models. We now discuss past research that we build upon, and use it to motivate our FSM and bisimulation approach.

RELATED WORK

Here, we discuss earlier efforts by others related to the topics of quantitative analysis of UIs, formal methods in HCI, model representation and simulation. Various approaches have aimed to improve user productivity through quantitative analysis of UIs [19]. Typical examples include *KLM* [5], *GOMS* [23] and *TAG* [28]. Such approaches require task analysis at a low level and hence are suitable when a user's behavior is either well understood [21] or error-free [7]. The inability of these approaches to deal with incomplete and imperfect user behavior [13] limits their utility for analysing user learning. We therefore propose a new approach to trace and measure user learning when defined as the progressive convergence of the gap between the target and the user model.

Mental models have long been in use in Learning and Cognitive Sciences [22]. Moreover, there is considerable debate on whether a mental model in general [27], or its FSM representation in particular, is in itself a sufficient description of what the user has in her mind. In our work, the novelty is not in using FSMs to represent mental models, but in representing the evolution of an individual's mental model during a guided learning process. Most works on guided learning are domain specific. *E.g.* Cognitive tutors [2] use separation of memory types into declarative and procedural to provide individualized support. However, learning an interactive system mostly involves procedural knowledge unlike learning algebraic concepts. We, therefore, focus on measuring a user's knowledge specific to learning an interactive system.

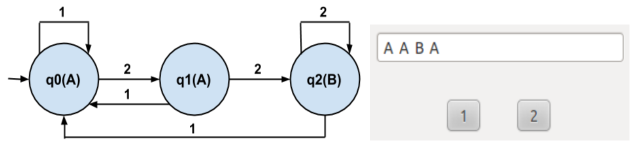
Formal methods have been widely used to specify, model and verify user interactions [16, 25, 32]. Several investigations have reported different methods for identifying and bridging the gap between what users know and what they need to know [4]. Systems engineering principles have helped to hide the irrelevant complexity of a machine from the UI [36]. Heymann *et al*. [18] used formal methods in proposing a systematic procedure for automatic generation of correct and succinct UIs. Several of the limitations of automatic UI generation have been described in detail [12, 29, 35]. Moreover, most of these techniques assume every user behaves in a similar and ideal manner, and therefore do not apply to novice users. This encourages us to propose a new technique to understand, evaluate and analyse individual user interactions with machines without the assumption of a generic user model.

The representation of a model is a key aspect that determines the effectiveness of any analysis technique. Different notations like maps, graphs and mathematical specifications have been used to represent user models [10, 15, 17, 30]. Some of these representations offer poor support for scalability and hinder a detailed analysis of the user models. Traces of user interactions with a machine form the primary source of data for user modeling and should ideally be analyzed as a time series. However, Dix *et al*. [1] have demonstrated how statistical and machine learning techniques fail to analyse such user traces but they can easily be applied if the representation of the behavior focuses on the state transitions. The formal language of automata has been used by others like [33] to represent both the target and the user model. This facilitates the use of mechanized techniques like model checking to automatically discover any scenarios that cause any divergence in the behaviors of the two models [34]. It also allows us to contribute towards safety and due diligence in safety critical interactive systems by identifying any instances in the machine that may lead to incorrect user understanding [38].

In our paper, we capture and compare the user models of an individual to understand the process of user learning. Milner used bisimulation to demonstrate behavioral comparison of two FSM models [26]. In [8], Combefis and Pecheur used a bisimulation-based equivalence relation to address the mode confusion problem. They demonstrate the possibility of generating user models for an imperfect user and take into account the deviations in her behavior. This motivates us to extend the idea of bisimulation-based equivalence to study how users learn a new machine by extracting, not one but multiple user models in sequence as the user interacts with the machine. To investigate the potential of our proposed idea, we conducted an exploratory study as described below.

EXPLORATORY STUDY

In order to investigate the process of how a user learns to use a machine, we conducted an exploratory study with 27 participants (17 males, 10 females) with an engineering background and captured the process of formulating user models for a machine which is unfamiliar to the user.



(a) Target model (b) User Interface

Figure 1. The target model and UI of the unfamiliar machine presented to participants during exploratory study and experimental study I.

Not surprisingly, users trust an unfamiliar machine quickly if it is simpler, due to their ability to predict the machine's behavior [39]. This trust in the machine's behavior assists users in breaking out of any incorrect user understanding and further learn the machine correctly. Therefore, we chose a simple machine (Figure 1) for our study. We labelled the states as q_0, q_1 and q_2 with observable outputs, *A* or *B*, alongside. The UI consisted of one output text box and two input buttons labelled *1* and *2*. The output display language consisted of letters *A* and *B*. As a subject pressed either of the two buttons, the output would display the previous history of outputs appended by either letter *A* or *B* following its deterministic machine behavior. These strings of letters *A* and *B* would remain in the text box over time.

Our chosen machine closely resembles a simple tea/coffee vending machine [26] that starts at state q_0, accepts a coin and makes a transition to state q_1, outputs tea, and returns to the initial state q_0. By accepting another similar coin in state q_1, it goes to state q_2 and outputs coffee before returning to state q_0. The machine does not change its state when a user inserts coins in state q_2. Also, it does not output tea or coffee without accepting any coins. In Figure 1, the transitions labelled *2* correspond to coin transactions, while the transitions labelled *1* correspond to a user accepting tea or coffee. Thus, we present an unfamiliar machine to the participants by introducing small changes into a familiar machine.

A confirmatory approach to extract, represent and analyze a user's understanding of a concept allows researchers to validate only a subset of a user's understanding [6]. We therefore chose the exploratory approach for this study. The unfamiliarity with the machine compelled the user to construct a model of her understanding in order to be able to successfully predict and use the machine. The users predicted the next character in sequence after each click of the buttons labelled *1* and *2* and thought aloud what they believed to be the behavior of the machine at regular intervals. Later, we attempted to represent the user knowledge gathered through verbal responses as FSMs. From our experiences in this exploratory study, we drew the following conclusions:

1. The users were able to perform similar tasks better when they interacted more with the machine. Therefore, we hypothesize that user learning can be represented as a series of user mental constructs.

2. The uncertainty within user responses to reason about a change in the machine made it difficult to translate well the user narratives of machine's behavior into an FSM. Therefore, we decided to recruit computer science literate partic-

ipants who could directly report their understanding as an FSM, rather than we infering it from a narrative.

3. Investigating evolution of user models can be of tremendous help for reasoning how users learn. Our proposed measure of user understanding should therefore allow for a rigorous analysis of the evolving user models, which may be incomplete *and/or* incorrect by nature.

These realizations helped us formulate our edit distance solution approach, as explained in the following section.

SOLUTION APPROACH: EDIT DISTANCE

Once the user model and the target model are formally captured and rendered, then learning can be operationally treated as the process of evolution of the user model towards the target model. In most cases, a user model deviates from the target model because of incomplete, as well as, incorrect learning. For a fine grained analysis of user learning, we need to quantify the gap between these two models. To measure the behavioral closeness of these models, we rely on the graph-theory technique of *edit distance*, a quantitative value which represents the gap. In our work, the user learning process is then hypothesized as a reduction in the edit distance: the progressive convergence of the gap between what the user believes to be the system's behavior (user model) and what truly is the system's behavior (target model). To address the concerns raised in our exploratory study, we used a bisimulation-based technique to calculate edit distance ensuring that our metric allows analysis of incomplete and incorrect user models. Next, we discuss the process of making a model bisimilar to another model by performing some edit operations.

Restricting edit operations to generate a bisimilar model

In general, an edit operation on an FSM can be any change in terms of deletion or addition of an edge or a node. But it is easy to observe that the presence or absence of an isolated node (without any incident edges) would not affect bisimilarity. Thus, deletion of a node simply corresponds to deletion of multiple edges modulo bisimilarity. Therefore, we do not consider removing a node as an edit operation. We define an edit operation on an FSM as deletion of an edge, addition of an edge or addition of a node with only one incident edge.

It is also easy to see that any two FSMs can be made bisimilar to each other by deleting all edges of one FSM and adding nodes and edges corresponding to the other FSM. This is a naive way of making two FSMs bisimilar. Furthermore, this would ensure structural (graph based) proximity and not behavioral proximity. We have had examples of user models from our experiments that were very structurally different from the target model but were behaviorally close to it. Therefore, we ensure that our chosen algorithm for edit distance modulo bisimilarity not only captures behavioral proximity as opposed to structural proximity, but also restricts the required edit operations using the following propositions.

PROPOSITION 0.0.4. *Let* $L_u = \langle X_u, q_{0u}, \delta_u, I, Y, H_u \rangle$ *and* $L_t = \langle X_t, q_{0t}, \delta_t, I, Y, H_t \rangle$ *be two FSMs. Let* $R : X_u \times X_t$ *be a relation such that if* $(q_{0u}, q_{0t}) \in R$ *and* $(s, q) \in R$*, then* $H_t(q) = H_u(s)\ \forall s \in X_u$*. Then there exists an FSM* L'_u *obtained by deleting only those edges of* L_u *that are not simulated by* L_t *via* R*, such that* R *becomes a simulation of updated FSM* L'_u *by* L_t*.*

PROPOSITION 0.0.5. *Let* $L_u = \langle X_u, q_u, \delta_u, I, Y, H_u \rangle$ *and* $L_t = \langle X_t, q_t, \delta_t, I, Y, H_t \rangle$ *such that* L_u *is simulated by* L_t *through a relation* $R \subset X_u \times X_t$*. Then there exists an FSM* L'_u *obtained by addition operations on* L_u *such that* L'_u *is bisimilar to* L_t *by the relation defined by* R *extended to the added states. Furthermore, the additions can be restricted to correspondence with only those transitions in target FSM not simulated by user FSM.*

We have described the formal proofs for both the propositions in detail in [3]. Proposition 0.0.4 says that a user model can be simulated by the target model by deleting a limited number (may be less than the total number) of edges from the user model. Further, Proposition 0.0.5 says that if a user model is simulated by the deterministic target model, then addition of a limited number (may be less than the total number) of nodes and edges to the user model would make the two models bisimilar. Based on this, we define the edit distance and hypothesize that it captures behavioral proximity. This hypothesis is validated through the experiments in the paper.

Calculating edit distance between 2 FSM models

Given a user model L_u and a target model L_t, and an arbitrary relation $R \subseteq X_u \rightarrow X_t$, there exists a model L'_u obtained by making the minimum possible number of edit operations on L_u such that R becomes a simulation of L'_u by L_t. Through the proof of Proposition 0.0.4 [3], it is easy to see that the model is unique to a relation defined. We can define multiple relations and for each relation there is a model. Let $S(L_u, L_t)$ denote the set of all such models. The minimum number of delete operations required to generate L'_u from L_u such that L'_u is simulated by L_t is denoted as $M_d(L_u, L_t, L'_u)$.

Now, there exists an L''_u, the FSM obtained by making addition operations on L'_u so that L''_u is bisimilar to L_t by an extension of the relation R to added nodes. This follows from Proposition 0.0.5. Denote $M_a(L'_u, L_t)$ as the minimum number of additions required to obtain one such model among all possible bisimilar models. Then, we define the edit distance between L_u and L_t as

$$e(L_u, L_t) = \min_{L'_u \in S(L_u, L_t)} [M_d(L_u, L_t, L'_u) + M_u(L'_u, L_t)] \tag{1}$$

Although it is computationally difficult to find the edit distance between two arbitrary FSMs due to the number of possible relations from states of one automaton to other, it may be found easily for small FSMs by associating the start states of the target and user FSM by the simulation relation and applying heuristic calculations. The catch here is the start states of the two FSMs normally correspond to each other. So, we take them as related states of the desired bisimulation relation. We illustrate the heuristic calculations with an example case taken from our experimental study I. Our example is described by the series of Figures 2, 3 and 4. We denote the user and target model in consideration as L_u and L_t in Figure 2 and find the edit distance between them. The model on

Start	Button	End	o/p
u0	1	u0	A
u0	2	u1	B
u1	2	u1	B

Table 1. State transitions of the target machine in Figure 2 as perceived by the user.

Start	Button	End	o/p
t0	1	t0	A
t0	2	t1	A
t1	1	t0	A
t1	2	t2	B
t2	1	t0	A
t2	2	t2	B

Table 2. Actual state transitions of the target machine.

the left is the user model and on the right is the target model. Each row of Table 1 and Table 2 respectively decribe the state transitions of the user model and the target model in Figure 2.

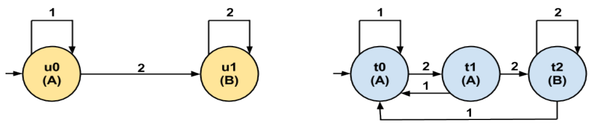


Figure 2. An example of a user model (left) and target model (right) taken from experimental study I.

The first step (Figure 3) in the process of calculating edit distance is to heuristically find a model L'_u by making a minimum number of edge deletions from L_u such that L_t simulates L_u. This is done by recursively associating states of L_u to those of L_t and deleting edges from L_u as necessary. The explanation follows. Initialize $L'_u = L_u$. The start state u_0 of L'_u should unequivocally be associated to the start state of L_t *i.e.*, $(u_0, t_0) \in R$. This association is correct with respect to the transitions $u_0 \xrightarrow{1} u_0$ and $t_0 \xrightarrow{1} t_0$ and hence we keep these transitions intact. Although $u_0 \xrightarrow{2} u_1$ and $t_0 \xrightarrow{2} t_1$, the output of u_1 (*i.e.*, B) and t_1(*i.e.*, A) are not the same (see Table 1 and Table 2). Hence, we do not map u_1 to t_1. Instead, we map u_1 to t_2 because $u_1 \xrightarrow{2} u_1$ and $t_2 \xrightarrow{2} t_2$ allowing the state t_2 to simulate u_1. This also mandates the deletion of the transition edge $u_0 \xrightarrow{2} u_1$ for preserving simulation of L'_u by L_t. This can be clearly seen in Table 1 that the user model is missing a state equivalent to state t_1 of the target model. The updated user model L'_u is now simulated by L_t and is obtained by a minimum possible number of edge deletions from L_u.

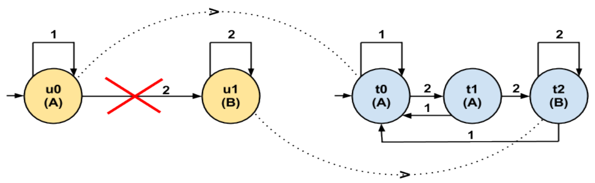


Figure 3. The simulation mapping of states of user model to those of target model and the requisite deletion operations from the user model such that the mapping enables the target model to simulate the user model.

In the next step as illustrated in Figure 4, because L'_u is simulated by L_t, we can get a L''_u bisimilar to L_t by adding edges or states with incident edges to L'_u. In the present case, a state u_2 is added to L'_u and also edges $u_0 \xrightarrow{2} u_2$, $u_2 \xrightarrow{2} u_1$, $u_2 \xrightarrow{1} u_1$ and $u_1 \xrightarrow{1} u_0$. Now after making L''_u and L_t bisimilar, we calculate the edit distance as the sum of the total number of deletions and the total number of additions $= 1+4 = 5$, as stated by equation 1. In the following sections, we describe two experimental studies we conducted to validate our proposed edit distance metric for user learning.

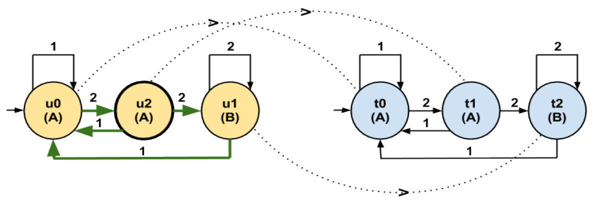


Figure 4. The required addition operations and the extension of the mapping to new states such that the new mapping is a bisimulation between the user model and the target model.

VALIDATION OF THE EDIT DISTANCE METRIC

Experimental Study 1

We setup a simple within-subjects experimental study to validate our proposed measure of user learning. We presented participants with a newly created unfamiliar machine and tasked them with learning its functionality by interacting with it. Each participant was routinely asked to self-report her impression of the functionality as an FSM, which represented her user model for that round. Over the course of an entire interaction, several rounds of user models and one final user model were captured. We compared this series of user models to the target model. Finally, we derived a trend by comparing each subject's stream of user models against the singular target model. The reliability of the proposed measure is confirmed by observing that the user models converge towards the target model with progressive learning of the user.

Participants

20 undergraduate/graduate students (16 males, 4 females), participated in the experiment. Their ages ranged from 21 to 43 years (average was 23). All participants were computer science literate comfirming their prior knowledge of FSMs.

Stimuli

The functional behavior of the machine, that was presented to the participants as a novel machine for learning, was modeled after the FSM described in Figure 1. The software also captured experiment duration, inputs provided, outputs generated, and the timing of such events. We used this information along with the self-reported user models for data analysis.

Procedure

We captured the demographic information of each participant and provided them with an overview of the experiment. A set of two questions were then asked: (1) Have you studied FSMs before? and (2) Do you feel comfortable to represent the behavior of a simple machine as an FSM? After obtaining her consent, the subject was lead into the experiment.

For the experiment, we initially trained the participant using a counter that allowed her to increase or decrease the displayed number by one ranging from *0* to *4* with the help of two buttons. The UI of the training machine was similar to the one described in Figure 1. The researchers helped the participant in representing her understanding of this training machine as an FSM to get the participant acquainted with (1) the UI of the machine, (2) the task that she was expected to perform and (3) how to represent an interactive behavior as an FSM.

After the training period, all participants were instructed to learn the functionality of the new machine, which was about to be displayed to them, by interacting with it. Operationally, each interaction meant clicking on the buttons labelled *1* or *2*, observing the corresponding outputs in the form of letters *A* and *B*, and studying the history of earlier outputs (*e.g.*, an accumulated text string of *AABABBA* ... in the display box).

The participants were informed that while they were interacting with the machine, they would be interrupted after every 3 interactions, each interaction being one click, and asked to draw an FSM that represented their understanding of the machine. This drawing served as their user model for that round. The subjects would also have to answer the question: "How well does your user model represent what is in your mind?" The possible answers were: (a) this is a very good representation of what I have in my mind, (b) this is a partial but a fair representation of what I have in my mind, and (c) this is an incomplete representation of what I have in my mind.

We allowed the participants to interact with the machine for any number of rounds. When the participant felt that she had a complete understanding of the machine, she could stop interacting and draw a final FSM, which was labelled as final user model (Uf) for that subject. To conclude the experiment, participants answered 3 final questions: (1) was this machine new to you? (2) when encountering a new machine, what do they think is more important: the number of interactions, *and/or* amount of time spent with the machine? (3) what is your preferred way of interacting with a new machine?

Results

All 20 participants had previously studied FSMs in their university curricula. In addition, 18 of them felt comfortable in using an FSM to represent their mental understanding of the machine. Most of the participants (18) felt that the machine was sufficiently new to them. Although the remaining 2 participants suspected a partial exposure to a similar device in their past, neither of them guessed the FSM correctly.

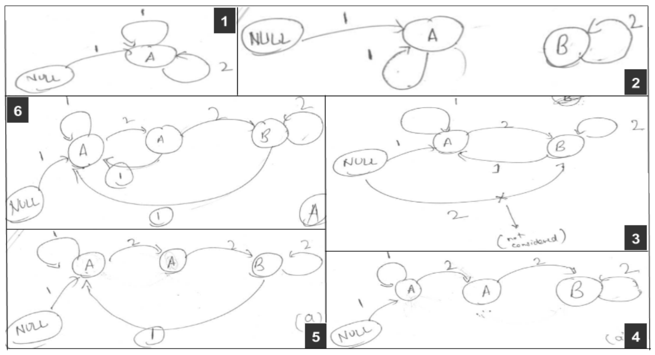


Figure 5. A sequence of user models self-reported by a participant during study I. Each model represents her understanding of the machine represented in Figure 1 at different rounds of interaction.

18 participants reported that their self-reported user model in the various rounds of the experiment was at least a good to fair representation of what was in their mind, with 10 even claiming it to be a very good representation. 13 of the 20 (65%) felt that when learning a new machine, it is the interactions with the machine (and not necessarily time spent with it) that is important for better understanding the machine. The remaining 7 felt that interactions *and* the time spent on the machine contribute to learning. However, when asked about how they have learnt about the functionality of a new machine in the past, 6 of these 7 said that they would consult a manual, and 1 said that he would look for a pattern. All agreed that they would indeed interact and explore the machine to learn it.

Our data revealed that every participant spent an average of 9:31 minutes in learning this new machine (maximum: 19:54, and minimum: 4:08). All interacted with the machine for at least 3 rounds; 15 completed 5 rounds, 7 completed 6 rounds, and only 1 person went up to round 8. The average time spent in each round was typically around 1:56 minutes.

The participants self-reported their user models as hand-drawn FSMs (Figure 5). The average number of user models per participant was 4.89 (SD=1.12). Almost all (19 out of 20) had the output symbols (letters *A* and *B*) as states, and input buttons *1* and *2* as the transition arcs. One participant chose to represent the state machine with multiple letters (as in A, AA, AB, *etc.*). Structurally, there were dissimilarities between the hand-drawn models across different users, but common features included A as the start state, at least 2 states, 4 transitions, *etc.* We computed the edit distance for each user model using our algorithm and plotted the trend against the multiple rounds of interaction for each participant in Figure 6. The overall convergence in the trend of edit distances validated our metric. A detailed comparison of the self reported user models and the resultant trend of edit distance can be found in the Discussion Section of this paper.

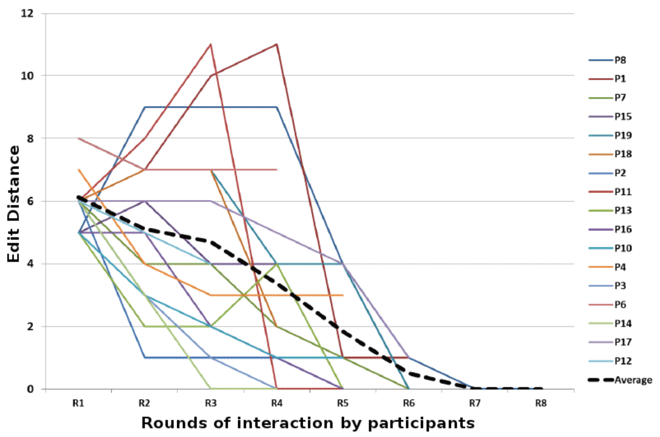
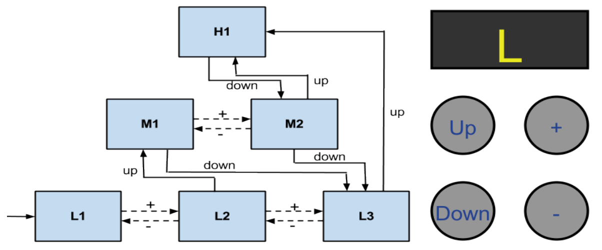


Figure 6. The trend of edit distance as it converges for different users across several rounds of user interaction. The dotted line indicates that the average of edit distances calculated for models of all users during the corresponding rounds of interaction shows a monotonic convergence.

Experimental Study 2

We conducted the previous study using a simple machine with low functional complexity. In order to further confirm the validity of our approach, we conducted another study involving a variation of a digital vehicle transmission system (VTS),



(a) Target model (b) User Interface

Figure 7. The target model and the UI of the digital vehicle transmission system (VTS) with variations to make it simpler. This machine was presented to participants of the experimental study II.

taken from [8]. VTS is a classic example of an interactive system that resembles a sufficiently complex system with limited transparency and issues of mode confusion.

Participants

The experiment involved twelve participants, 9 males and 3 females, with ages ranging from 18 to 62; average age was 27.25 with a standard deviation of 11.69.

Stimuli

We presented the participants with a variation of a VTS model, taken from [8], as an unfamiliar machine. The participants interacted with this machine described in Figure 7, that represents the target model and the new UI for this study.

Procedure

The protocol for this study remained largely similar to the previous study, except for our approach for extracting the user models. From the different techniques available for extracting user models, such as structured interview, questionnaire and talk back, an open method is less useful than a more structured approach when the complexity of a machine increases [31]. The digital VTS model presented to the participants, even with the variations meant to simplify it, was sufficiently complex for an unfamiliar user and necessitated a structured approach to extract user models. Therefore, we provided partial guidance to the study participants in identifying the states of the target model. The participants received this guidance in the form of sheets of paper having a minimum of all the states, which were necessary to represent the target model, printed upon them. They were expected to use these sheets to self-report their understanding of the machine, thus reducing the task of drawing FSMs to identifying the valid states and mapping the transitions between them. This helped the researchers to minimize the noise captured within a user model due to a participant's inefficacy, while reproducing her understanding as an FSM. We provided a few superfluous states on the sheets to (1) ensure that particpant were not influnced by the number of states marked on the sheets and (2) allow them to express their understanding as an FSM that may be structurally different, but bisimilar to the VTS target model.

While conducting the study, we started by collecting the demographics of the participant and her prior knowledge about FSMs and her ability to draw one. Since the target model was a variation of a VTS, we also enquired if the participant had any experience with using a car gear box. The participant was shown a simple FSM with 2 states and 1 transition, and was asked to think aloud to explain it. After ensuring a correct understanding of the concept of an FSM, the participant received training on how to represent an interactive behavior as an FSM. We used the same training example of a number counter, taken from the first study.

The participants were provided with a brief explanantion of a VTS model having a few main gear levels and a few sub-levels within each of the main gear levels. We informed them that we would be using a variation of a VTS with 3 main gear levels: low, medium and high, and each level would have a minimum of 1 and a maximum of 4 sub-levels. However, the software would only display a character corresponding to the main gear level (*i.e.* L, M or H), irrespective of the sub-level. It was possible to separately identify all the sub-levels within a main gear level through sufficient interaction.

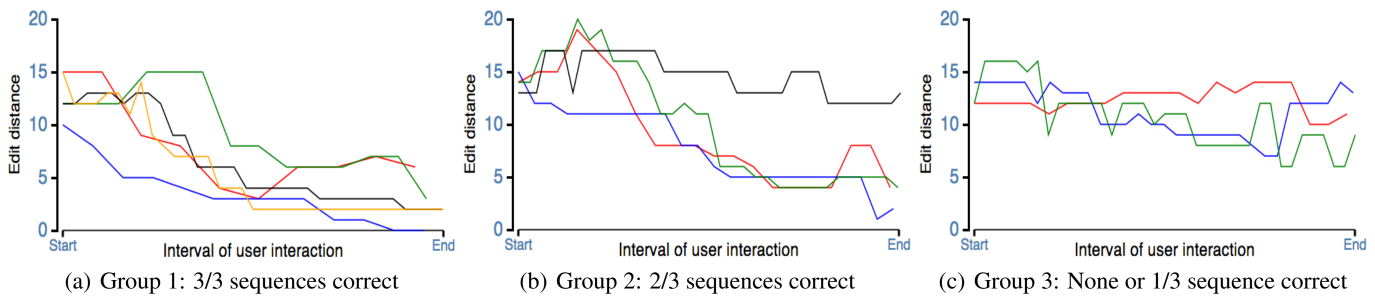
The participants were tasked to learn the behavior of the machine (Figure 7) by interacting with it for an unlimited number of rounds, where each round consisted of four clicks. They were free to distribute the 4 clicks in each round amongst the 4 buttons on the UI (*i.e.*, Up, Down, + and -) as they saw fit. Then, they were asked to self-report their understanding of the machine as an FSM, one on each sheet of the paper with the possible states indicated on them. We notified the participants that the most succint representation of the VTS would not take more than 9 states; but they were free to utilise all 16 states shown on the provided sheets.

A round of interaction was complete when a participant answered the following questions about her user model for that round: (1) What fraction of the machine's behavior did she understand till then?- *(a)* 0-25% *(b)* 25-50% *(c)* 50-75% *or (d)* 75-100% and (2) How well did the FSM represent her understanding of the machine?- *(a)* poor *(b)* fair but partial *or (c)* fair and complete. The response of the former question reflected the gain, if any, in participant's understanding of the machine; while the latter reflected the efficiency and completeness of her representation, similar to the first study.

At the end of the second and the final round of interaction, we tested the participants' understanding. We asked them to indicate, using the smallest number of clicks, how to transition between two states of the model, *e.g.*, start state of the machine to any of the medium sub-levels. These responses reflect if the participant learnt anything during the several rounds of interaction or not. Each participant answered the same set of 3 final questions, taken from the previous study, after her interaction session with the machine was complete.

Results

9 (out of 12) participants had studied FSMs and thought they would be able to plot an interactive behavior as an FSM. Except one, all participants had an average of 6.1 years driving experience (SD=12.6). The brief explanation of a VTS model (prior to the study) ensured that each participant had at least some knowledge of how a car gear box works. Despite this, 11 participants confessed that the VTS model was unfamiliar to them; this included a participant who had a thorough



(a) Group 1: 3/3 sequences correct (b) Group 2: 2/3 sequences correct (c) Group 3: None or 1/3 sequence correct

Figure 8. The edit distances for all user models, computed using our algorithm, collected in the experimental study II performed using the vehicle transmission model. We have grouped the users as per their final understanding of the machine, reflected by the number of correct user responses. X axis denotes the normalised duration of time that each participant spent trying to learn the machine and actually made any changes to her user model.

knowledge of the internal strucure and working of a car gear box. One of the participants, with prior driving experience, reported the behavior to remotely resemble a game to launch a rocket at different speeds depending on the player interaction with the knob to release the rocket. She also tried to relate it to a new type of clutch that decides how to shift gears, depending on how hard it is pressed.

5 participants reported the number of interactions allowed with the machine to be the primary reason for them to understand the behavior of the VTS model, while 6 participants gave an equal priority to number of interactions and the time to think between them. One participant found her knowledge of the previous state where he left the machine to be more important. When questioned about the preferred way to understand the working of an unfamiliar machine, 9 preferred to interact with it; 2 preferred using a manual, while one said that she would interact with a simple machine but refer to a manual if the machine were sufficiently complex.

The participants interacted with the machine for an average of 28.5 rounds each (minimum 10 and maximum 43), with a standard deviation of 11.04. In the end, all the participants reported having understood most of the machine (reported as 75-100%) and were confident about their representation to be fair and complete, 5 participants stopped interacting once their model was complete, while 7 continued to interact further and used their final few rounds to confirm their previous understanding as represented by their last FSM. The participants took an average of 55:30 minutes (SD=21:30 minutes) to interact with the machine before they thought they understood it completely; the minimum was 27:37 minutes while the longest was 108:57 minutes.

We classified the users depending on the correctness of their responses (required sequence of clicks) to perform certain state transitions as requested by the researchers into 3 groups. We plotted the edit distances for user models, as computed by our algorithm, across the normalised duration of time that each participant spent trying to learn the machine, separately for each group in Figure 8. The normalised duration excludes the time spent by participants in past few rounds for verifying their previous knowledge of the machine during which they neither changed their user model, nor their confidence about their understanding of the machine or their representation of it. A detailed discussion of these follows in the next section.

DISCUSSION FOR BOTH EXPERIMENTS

By using the self-reported user models in both studies as our primary source of data, we trace user learning and measure the gaps between the user models and the target model in progressive rounds of interaction. We quantify this gap using our proposed measure edit distance. We then compute the edit distances for each of the user models per participant, and plot them against their corresponding rounds of interaction as observed during study I in Figure 6. The simplicity of the machine used in study I ensured that all the participants learnt some fraction of the machine's behavior through user interaction. Considering the complexity of the VTS model, we classified the participants of study II into 3 separate groups to facilitate a better analysis of the results as shown in Figure 8: participants who answered all 3 understanding questions correctly, answered 2 correctly, and 1 or none correctly.

The graph in Figure 6 clearly indicates that the overall trend of edit distances converges with every passing round as a user interacts more with the machine. A similar trend of convergence can be seen in Figure 8a. This graph reflects how edit distance converges for a user who learns the VTS model correctly (*i.e.*, answers all understanding questions correctly). 5 participants from experimental study I learned the machine partially. This is captured as a converging but non-zero edit distance. A similar convergence can be seen in Figure 8b (except for one user) despite their inability to perform all the given tasks correctly. Our measure successfully illustrates their process of acquiring this partial understanding.

As Figure 6 indicates, a few participants in study I have their trend of edit distances higher than the average. On examining their user models, we realised that initially the users who contributed to the 2 highest peaks, struggled with the concept of FSMs since they could not distinguish between the outputs and the states of the machine and resorted to plotting their user models as a sequence of outputs as opposed to an FSM. We also notice an outlier in Figure 8b showing no significant convergence in edit distance despite providing correct user responses to our state transition questions. A further probe into her user models indicated that the she identified behaviorally similar states as separate without mapping all the necessary transitions between them to ensure bisimilarity. The replication of behaviorally similar states in a user model, in turn replicates the number of missing transitions, if any. Our edit distance algorithm includes each missing transition and outputs a higher edit distance indicating user confusion. We consider that a user with a partial understanding and a larger number of states in her model is more likely to be confused than a user with similar understanding but fewer states. We suspect this to indicate higher cognitive load. Our metric thus identifies these cases as partially incorrect user learning.

The next peak in Figure 6 comes from a participant who demonstrated an erroneous learning of the machine thus justifying an increase in the edit distance towards round 2. This participant's user models indicate that she recreated her user model following round 4 when she realised her error, as is evident from the steep decline in her edit distance plot indicating a positive knowledge gain. Similar instances of erroneous learning can be seen in trends of several users during study II. The rise in edit distance for one participant in Figure 8a resulted from an erroneous learning which was reflected in her correct, but not the shortest, sequence of clicks for the third task. Figure 8c especially highlights cases where users either could not rectify their erroneous learning or were unable to learn constructively about the VTS model. The absence of any convergence in this graph implies no user learning, thus further indicating the reliability of our proposed edit distance.

Thus, our proposed measure edit distance successfully traces cases of complete, partial, erroneous and no user learning. These trends will allow us to identify users that need some help in learning an interface or UIs that need to be redesigned. In the future, this approach can be used to investigate the rate at which a user attains her understanding of an unfamiliar device by investigating how fast do the user models converge towards the target model. Our proposed technique of edit distance is useful for allowing a behavioral comparison between any two models that can be represented as FSMs. *E.g.* comparing different UIs for the same system, comparing business process models as they evolve with time, *etc.*

Ideally, edit distance would be the minimum number of edges and states with incident edges required to be added and deleted from the user model such that it bisimulates the target model. However, it would be computationally very expensive to compute the edit distance by an automated algorithm due to a large number of possible relations between the states of the user model and the target model. The heuristic edit distance that we computed manually using our algorithm, therefore, may vary from the ideal edit distance and compromise its minimality; however it is logically and scientifically consistent, as shown in Proposition 0.0.4 and Proposition 0.0.5.

A non-deterministic user model can certainly be made bisimilar to the target model by using a naive $O(|G1| + |G2|)$ algorithm, which essentially deletes G1 edges and reconstructs G2 edges. However, such an edit distance is not minimal. The problem of computing the minimum edit distance is NP-Hard. On the other hand, efficiently finding the minimum edit distance between two non-deterministic FSMs modulo bisimilarity is an open problem. While not optimal, our present algorithm performs better than the naive $O(|G1| + |G2|)$ algorithm and measures the edit distance between two deterministic models. This restriction to deterministic user models is not a theoretical limitation of our work; instead it prevents the user from expressing ambiguity by reporting only deterministic approximations of the machine, which the user was informed to be deterministic.

Although we have defined (bi)simulation in the context of FSMs, bisimulation is a general concept used to capture the notion of behavioral equivalence and extends across different domains *e.g.* [11]. Thus, the idea presented in the paper can be applied even when user interactions are represented in different notations and is not restricted to FSMs.

CONCLUSION

The paper focuses on a behavioral comparison of user and target models when represented as finite state machines. We propose the *edit distance* metric for mathematically capturing the gap between evolving user models and a target model using bisimulation equivalence. In our experiments, we observed that the edit distance between the user and target model converges with progressive user learning, increases for erroneous learning and remains unchanged indicating no learning. Thus, the paper demonstrates that user learning can be witnessed, captured, and measured formally, allowing for a better understanding of how users learn to use a new device.

Our proposed edit distance metric is of significant importance to UI designers to identify instances when user understanding is not improving and to develop dynamic adaptive interventions to address them [9]. A quantified measure of user learning will help researchers to develop techniques that expedite the process of training a user for successfully operating a new device. In the future, we plan to use our proposed framework for segmentation of the feature space of a device depending on which features a user could understand completely or partially, and those, which he could not use at all. Ensuring complete knowledge of some features will be crucial to safety critical systems for meeting minimum standards of safety. Such an effort will also help in identifying new design principles leading to better user interaction.

REFERENCES

1. A. Dix, J. F., and Beale, R. Analysis of user behaviour as time series. In *HCI'92: People and Computers VII*, Cambridge University Press (1992), 429–444.
2. Anderson, J. R., Corbett, A. T., Koedinger, K. R., and Pelletier, R. Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences 4*, 2 (1995), 167–207.
3. Arvind, S., and Choppella, V. A definition of edit-distance modulo bisimulation. Tech. Rep. IIIT/TR/2014/1, IIIT-Hyderabad, January 2014.
4. Baecker, R., Booth, K., Jovicic, S., McGrenere, J., and Moore, G. Reducing the gap between what users know and what they need to know. In *CUU '00*, 17–23.
5. Card, S. K., Moran, T. P., and Newell, A. The keystroke level model for user performance time with interactive systems. *Commun. ACM 23*, 7 (1980), 396–410.
6. Carley, K., and Palmquist, M. Extracting, representing, and analyzing mental models. *Social Forces 70*, 3 (1992), 601–636.

7. Carroll, J. M., and Olson, J. R., Eds. *Mental models in human-computer interaction: research issues about what the user of software knows*. National Academy Press, Washington, DC, USA, 1987.

8. Combéfis, S., and Pecheur, C. A bisimulation-based approach to the analysis of human-computer interaction. In *EICS '09*, ACM (2009), 101–110.

9. Dhawan, R., M. O., and Borman, M. Mental models and dynamic decision making: an experimental approach for testing system methodologies. In *24th International Conference of the System Dynamics Society* (2006).

10. Doherty, G., Campos, J. C., and Harrison, M. D. Representational reasoning and verification. *Formal Aspects of Computing 12* (1998), 260–277.

11. Dong, Q. A bisimulation approach to verification of molecular implementations of formal chemical reaction networks, 2012.

12. Falb, J., Popp, R., Rock, T., Jelinek, H., Arnautovic, E., and Kaindl, H. Fully-automatic generation of user interfaces for multiple devices from a high-level model based on communicative acts. In *HICSS* (2007), 26–26.

13. Fischer, G. User modeling in humancomputer interaction. *User Modeling and User-Adapted Interaction 11*, 1-2 (2001), 65–86.

14. Gentner, D., and Stevens, A. *Mental models*. Cognitive Science - Lawrence Erlbaum Associates. Lawrence Erlbaum Associates, Incorporated, 1983.

15. Harrison, M. Modelling user structures within system specifications. In *Formal Methods in HCI: III, IEE Colloquium on* (1989), 1/1–1/4.

16. Harrison, M., and Thimbleby, H. *Formal Methods in Human Computer Interactions*. Cambridge Middle East Library. Cambridge University Press, 1990.

17. Hermann, M., and Weber, M. When three worlds collide: a model of the tangible interaction process. In *OZCHI '09*, ACM (2009), 341–344.

18. Heymann, M., and Degani, A. Formal analysis and automatic generation of user interfaces: approach, methodology, and an algorithm. *Human Factors 49*, 2 (Apr. 2007), 311–30.

19. Hinckley, K., Cutrell, E., Bathiche, S., and Muss, T. Quantitative analysis of scrolling techniques. In *CHI '02*, ACM (2002), 65–72.

20. Ippel, M. J., and Beem, A. L. Mental models as finite-state machines: Examples and computational methods. Tech. Rep. 9911, United State Air Force Armstrong Laboratory, October 1998.

21. John, B. E., and Kieras, D. E. The goms family of analysis techniques: Tools for design and evaluation. Tech. rep., Carnegie Mellon University, 1994.

22. Johnson-Laird, P. N. *Mental models: Towards a cognitive science of language, inference, and consciousness*, vol. 6. 1983.

23. Kieras, D. E. Towards a practical goms model methodology for user interface design. *Handbook of human-computer interaction* (1988), 135–158.

24. Kieras, D. E., and Bovair, S. The role of a mental model in learning to operate a device. *Cognitive Science 8*, 3 (1984), 255 – 273.

25. Long, J., and Dowell, J. Formal methods: the broad and the narrow view. In *Formal Methods and HCI: II, IEE Colloquium on* (1988), 5/1–5/8.

26. Milner, R. *A Calculus of Communicating Systems*. Springer-Verlag New York, Inc., 1982.

27. Norman, D. Some observations on mental models. *Mental models 7* (1983).

28. Payne, S. J., and Green, T. R. G. Task-action grammars: a model of the mental representation of task languages. *Human-Computer Interaction 2*, 2 (June 1986), 93–133.

29. Popp, R., Falb, J., Raneburger, D., and Kaindl, H. A transformation engine for model-driven ui generation. In *EICS '12*, ACM (2012), 281–286.

30. Qian, X., Yang, Y., and Gong, Y. The art of metaphor: a method for interface design based on mental models. In *VRCAI '11*, ACM (2011), 171–178.

31. Read, J. C., MacFarlane, S., and Casey, C. What's going on?: discovering what children understand about hand-writing recognition interfaces. In *IDC '03*, 135–140.

32. Reeves, S. Principled formal methods in hci research. In *Formal Methods in HCI: III, IEE Colloquium on* (1989), 2/1–2/3.

33. Romera, M. E. Using finite automata to represent mental models. Master's thesis, San Jose State University, 2000.

34. Rushby, J. Using model checking to help discover mode confusions and other automation surprises. In *Reliability Engineering and System Safety*, vol. 75 (2002), 167–177.

35. Schneider, K., and Cordy, J. Abstract user interfaces: A model and notation to support plasticity in interactive systems. In *Interactive Systems: Design, Specification, and Verification*, vol. 2220. Springer, 2001, 28–48.

36. Thimbleby, B., Blandford, A., Cairns, P., Curzon, P., and Jones, M. User interface design as systems design. In *People and Computers XVI - Memorable yet Invisible*, Springer (2002), 281–301.

37. Thimbleby, H. *Press On — Principles of Interaction Programming*. MIT Press, 2007.

38. Thimbleby, H. Contributing to safety and due diligence in safety-critical interactive systems development by generating and analyzing finite state models. In *EICS '09*, ACM (2009), 221–230.

39. Tullio, J., Dey, A. K., Chalecki, J., and Fogarty, J. How it works: a field study of non-technical users interacting with an intelligent system. In *CHI '07*, 31–40.

40. Zade, H., and Choppella, V. Functionality or user interface: Which is easier to learn when changed? In *IHCI '12* (2012), 1–6.