Controlling simulations of human-artifact interaction with scenario bundles

WILHELM F. (WILFRED) VAN DER VEGTE, ZOLTÁN RUSÁK
Delft University of Technology
Faculty of Industrial Design Engineering
e-mail: {w.f.vandervegte,z.rusak}@tudelft.nl

Keywords: Scenarios, product design, virtual prototyping, hybrid simulation, use process.

Abstract: *We introduce a methodology for modeling and simulating fully virtual human-artifact systems, aiming to resolve two issues in virtual prototyping: (i) integration of distinct modeling and simulation approaches, and (ii) extending the deployability of simulations towards conceptual design. We are going to offer designers a new way of investigating the use of a product, by integrating scenarios of expected human-artifact interaction and simulations of artifact behavior into a unified framework. The proposed simulation method is fully virtual, which is an advantage if recruitment and employment of human subjects for physical and virtual testing is problematic. The models incorporate both logical and physical aspects of the behaviors of humans and artifacts. This paper elaborates on the logical modeling and simulation elements, which are used to create scenario bundles that capture multiple possible ways of how virtual users interact with products, and represent the control that humans exert during interaction. We will present an outline of the fundamental theory and a pilot implementation that we applied for applicability testing to obtain a proof of the concept. We found that within limitations imposed by the commercial software we used, we could run simulations of virtual human-product interaction during the use of a product with sufficient fidelity. These simulations involved various basic human-artifact interactions, such as reaching, operating a button, and grasping. They provided useful knowledge on the improvements needed to develop a full-fledged dedicated simulation package, which will eventually offer designers the possibility to model scenario bundles and run simulations to investigate interactions with variations of concept designs and of interaction parameters.*

1. INTRODUCTION

This paper presents a solution for control of computer-supported behavioral and interaction simulations based on the concept of scenario bundles, with special attention to the conceptual design of consumer durables. A scenario bundle is a formalized description of a set of scenarios for the use of a product. The problem is of specific importance because including scenario-based simulations offer not-yet unleashed potential to provide designers with quantitative feedback on complete interaction sequences that can happen during product usage, and thus to provide clues to designers on how they can improve product designs.

With a scenario bundle, a designer can perform 'what-if' type of studies involving variations in the product's design, in its physical properties, in the surroundings of use, and in human users. Each time a new variation is introduced, the simulation may take a different course through the scenario bundle.

We consider this a useful complement to conventional engineering simulations, which require separate disconnected simulation runs for each specific interaction situation that can occur during the use process. Additionally, scenario-based simulation can be a low-threshold alternative to interactive simulation and to testing of physical prototypes, since there is no need to employ human subjects.

Originating from software engineering [1], the concept of scenarios has become widely used in various application fields of design. We adapted a definition of a *scenario* from [2] to include unintended behaviors (such as failure): a scenario is *a possible way of how a human user controls his or her interaction with a given product in given surroundings*. The 'way' of using a product refers to the different decisions a human can take that influence the course of the use process. The decisions control how the user interacts with the product. In order to be able to simulate scenarios, we must simulate this control.

Scenarios exist as informal and formal models [3]. The most common application of scenarios in product design is as informal descriptions to explore possible ways of, and to communicate preliminary ideas about, product usage [4]. Formalized scenarios have been used in computer-based simulations of human-artifact systems, but these simulations are typically based on drastically simplified interpretations of human-artifact interaction systems, for instance by not including direct physical interaction between humans and artifacts [5], or they do not support the possibility of various different interaction sequences (e.g., [6])

To make scenario-based simulation useful for product designers, our goal has been to overcome these limitations. We have hypothesized and developed a solution by building a proof-of-concept implementation with commercially available software packages. It was tested by modeling and simulating the use of a conceptual product. The applicability testing proved that scenarios could be simulated as hypothesized. This strengthened our belief that, if it is further developed to a full-fledged dedicated system, our methodology of interaction simulation based on scenario bundles can be an efficient approach for considering alternative uses and use processes in early virtual prototyping.

In section 2, we will outline the background knowledge we used and evaluate work that others have done in this area. In sections 3-5 we will introduce and explain the fundamental concepts of human-artifact interaction simulation controlled by scenario bundles. Section 6 discusses the development of the pilot implementation that is applied to an example product. Section 7 discusses the simulation results, and finally, in section 8 our conclusions and suggestions for further research are presented.

2. BACKGROUND KNOWLEDGE AND RELATED WORK

Considering the context of what others have already done to represent and simulate multiple use processes based on scenarios that define the possible ways of how humans control their interaction with artifacts, we will first investigate the background literature on representing use processes as scenarios in 2.1. Then, in 2.2 we investigate theories to capture human control behavior in models, and that can be used in simulations. Finally, in 2.3, existing approaches of scenario-based human-artifact interaction simulation are reviewed to learn what is currently missing.

2.1. Representing use processes based on scenarios

Our goal is to simulate multiple ways of how a product can be used. To make this possible, we need to resolve the issue of finding a common carrier for all these 'ways'.

As we defined in the introduction, a scenario is a possible way of how a human user controls his or her interaction with a given product in given surroundings. A widely accepted interpretation of scenarios in the context of product use is based on a theory of human problem solving introduced by Newell and Simon [6]. In this theory, a decision tree describes the possible actions one human can take towards a given goal, when controlling his interaction with his environment. The application of this theory to the use of products has been elaborated by Stanton and Baber [8], who state that the goal of use is to reach a solution to a problem. They describe the problem-solving task as similar to moving through a maze, from the initial state to the goal state. Each junction has various paths representing state-transforming operations. From each junction the user selects one operation, its execution causing a change of state. In the context of user-product interaction, each of the possible routes through such a network of options is commonly known as a particular scenario of use, as Hsia et al. phrased it [9]. Like Newell and Simon, Hsia et al. used a tree to represent the network, and called it a scenario tree. Since the tree representation is just a particular way of grouping scenarios, we will call any multiple-scenario representation a *scenario grouping.*

Our goal has been to find a representation form for multiple scenarios as paths of state-transforming control operations, which can be used in simulations and which is based on background knowledge from scientific research on human control behavior. In the next subsection, we examine existing scientifically underpinned models of how humans exert control on their environment (which includes artifacts or products).

2.2. Theories on modeling human control behavior

The theory of how humans control their interactions with their environment is studied in the field of human motor control, which covers a wide variety of activities such as walking, looking, reaching, grasping, drawing, keyboarding, speaking, etc. [10]. The primary human subsystems involved in this control behavior are the brain and the central nervous system. The brain receives input from receptors (i.e., sense organs) through the central nervous system, and provides output to effectors (i.e., muscles) through the central nervous system. We take this assumption as a starting point. Two prevailing viewpoints emerge from the investigated literature, namely, the information-processing theory, which is based on discrete control models, and the proportional-control theory, which is based on continuous control models.

The theory of human information-processing [11] has become widely accepted in the science of human motor control (e.g., [12]) and in cognitive psychology (e.g., [13]). The human is considered and modeled as a processor of information, comparable to a computer, and has receptors, effectors, and an intervening control system, with information processing concerned primarily with the operations of the control system [14]. Signals from the receptors (i.e., sense organs) are processed by the control system through the following sub-processes: (1) detection, (2) recognition, (3) decision-making (4) response selection, and (5) response execution. Response execution controls the motion of effectors. Decision-making is typically considered the 'highest level' of control. As can be expected based on the computer analogy, the models used in this approach are based on discrete-time processing of logic.

The theory of proportional control [15] has its foundations in the theory of classical control systems, where control behavior is described by differential equations and Laplace transforms. The focus area of this theory corresponds to response execution in the theory of information processing that we described above. The main goal of the control models is to calculate forces that muscles need to exert based on positions, angles, velocities, and angular velocities planned by the brain. Costello [16] reserves proportional control for modeling small corrections, while large-scale movements are represented by discrete-time models.

Indeed, findings over the past decennia confirm that the human brain controls movements based on positions, angles, velocities, and angular velocities rather than on forces and accelerations (e.g., [17]). Control of velocity and position is generally considered not to be based on continuous signals but on discrete, pulsatile signals [18].

From these findings we conclude that for most human control behavior, models that conform to the information-processing approach (i.e., discrete-time, logic-based models) are to be preferred over models based on proportional control. However, when simulation of detailed muscle movements requires prescribed forces, a proportional-control based model is needed.

2.3. Existing approaches for scenario-based simulation of human control

Several scenario-based simulation approaches for use processes have been proposed from the 1990s onwards. Focusing on human control behavior, we investigated these approaches to establish what is currently missing. The simulation models we found in the literature are typically based on known formal logical representations, such as Petri nets or statecharts. We will first discuss the existing approaches with a focus on which aspects of control are simulated. The formal representations, some of which are used in more than one approach, are then discussed in the next subsection.

The current approaches can be subdivided into two categories according to the field of application. One is that of simulations of human-computer-interaction (HCI), in which the artifact (product) is a software program and the focus is on data and information exchange between the human and the artifact. The other category is that of simulating human-artifact interaction in a virtual physical environment. In these approaches, physical interaction is simulated between the human and the product and/or between the product and other artifacts in its environment.

Scenario-based simulations in human-computer interaction: Although numerous HCI publications propose formal representations for scenarios and scenario groupings, only few are actually used to control simulations of use processes. The various scenario representations are typically used in requirements specification and in verification. One of the few applications to simulation is presented by Rauterberg et al. [19]. They use Petri nets to represent scenario groupings that simulate decision-making by a virtual human, who is interacting with a software product. Figure 1 shows which subsystems of the human and of the product are simulated. In software engineering, formalized scenarios represent human decision-making. Simulations involve processing of information by the human and by the product, as well as the exchange of information. Since physical interaction and physical behavior is ignored, human control is reduced to decision-making.

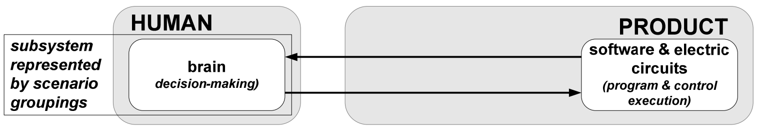

HUMAN
PRODUCT
subsystem represented by scenario groupings
brain
decision-making)
software & electric circuits
(program & control execution)

Figure 1. *Current usage of scenario groupings in computer-based simulations of HCI as proposed by Rauterberg et al. [19]. The arrows represent information flows (i.e., control signals). The 'product' is computer software.*

Scenario-based approaches involving physical interaction simulation: An increasing number of approaches are being proposed for simulation of human-artifact interaction in a virtual physical environment. In the Iowa driving simulator, Cremer et al. [5] used statecharts to model scenario groupings of virtual-driver control behavior. Contrary to the approach depicted in Figure 1, scenario groupings do not only cover human decision-making, but also control within artifacts. Physical behavior concerning interaction between the product and its surroundings (together addressed as 'artifacts' in the figure) is simulated, but not the physical human-artifact interaction. As a result, only the end effect can be studied in the simulation as behavior of the car interacting with its environment (Figure 2). For the physical interaction between the car and its environment, rigid-body mechanics simulation algorithms are used, which are co-simulated with the logic. Filla [20] applied a similar approach to virtual prototyping of a virtual wheel loader controlled by a virtual human driver that performs various tasks in a

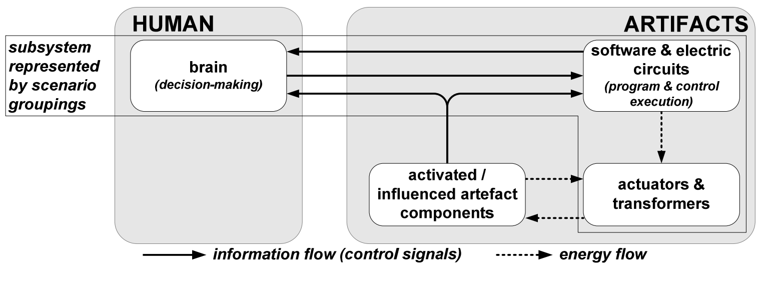

HUMAN
ARTIFACTS
subsystem represented by scenario groupings
brain
(decision-making)
software & electric circuits
(program & control execution)
activated / influenced artefact components
actuators & transformers
information flow (control signals)
energy flow

Figure 2. *Usage of scenario groupings in computer-based simulations as proposed by Cremer et al. [5] and by Filla [20]. 'Artifacts' refers to the product on which the simulation focuses (e.g., car) and to other artifacts in its environment.*

virtual environment. A more human-centered approach was recently proposed by Honglun et al. [6]. It includes control of human body parts that physically interact with the artifactual environment. In addition, the behavior of sense organs is included in simulations (Figure 3). A scenario is represented as a linear sequence of textual commands describing decision-making. It is not explained how the lower levels of control are included. What is missing in this approach is a possibility to consider multiple scenarios, as well as the possibility to include control within artifacts.

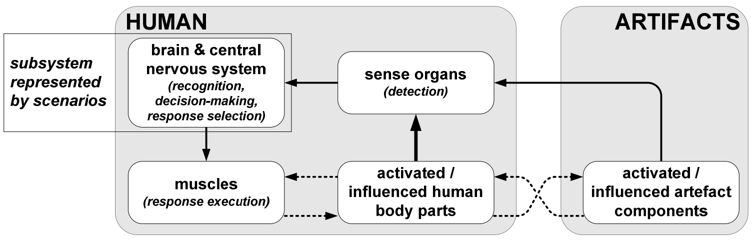

HUMAN
ARTIFACTS
subsystem represented by scenarios
brain & central nervous system
(recognition, decision-making, response selection)
sense organs
(detection)
muscles
(response execution)
activated / influenced human body parts
activated / influenced artefact components

Figure 3. *Usage of scenarios in computer-based simulations as proposed by Honglun et al. [6]. Legend: see Figure 2.*

Summarizing the above findings, Figure 4 brings together the subset of the interaction behaviors covered by the existing scenario-based simulation approaches, completed with detection behavior in artifacts. A comparison of this figure with the preceding three figures reveals what the existing approaches lack. Our conclusion is that a simulation approach that can comprehensively cover use processes should cover the behaviors of all the human and artifactual components in the figure, and cover all the interactions (interfaces) between these components. To make simulations scenario-based in accordance with our purposes and our definitions, it must be possible to define scenario groupings that represent human control behavior only, as depicted in Figure 4.

3. INTRODUCTION TO THE FUNDAMENTAL CONCEPTS

In this section, we will introduce our concept of controlling interaction simulations with the scenario groupings we developed for this purpose, and which we have called *scenario bundles*. As was stated in 2.2, the goal is to cover all the behaviors (blocks) and interfaces (arrows) shown in Figure 4. The scenario bundle that is to represent human control behavior is simulated by using discrete-time logical models rather than calculus-based continuous-time models. An exception is low-level force control, for which continuous simulation appears to be preferable.

In order to enable comprehensive investigation of use processes, the physical processes that are controlled, and which are connected by solid arrows in Figure 4, must be simulated concurrently. Co-simulation of discrete-time and continuous-time processes is commonly known as hybrid simulation [21]. Physical processes are typically simulated with physics models that have been developed for continuous simulation [22]. Examples of such models used in product design are finite-element models and rigid-body 3D volumetric models [23]. The continuous part of the co-simulation is not the subject of this paper. Our proposed solution for this has been presented in [24], where we also show how low-level force control can be incorporated in the continuous simulation, and can thus be left out of the scenario bundle.

Logical representations have been developed for various purposes. The main two categories of representations used in human control modeling are production rule systems (e.g., [25]) and automata. We use ‘automata’ to denote all kinds of graphical and/or matrix representations that describe logical behavior as groupings of states and transitions between states. This includes finite state machines and state transition diagrams [26], Markov models [27], statecharts [28], modecharts [29], and Petri nets [30]. We chose to use the statechart representation, which has proven itself to be suitable for modeling logical behavior in simulation of human-product interactions [5], [20].

Two key features missing in statecharts that are offered by other representations are probability (by Markov models and stochastic Petri nets) and timing (by timed Petri nets and by modecharts). In a scenario bundle these features can be interesting for modeling uncertainty in human behavior and for modeling human latency (e.g., hesitation), respec-

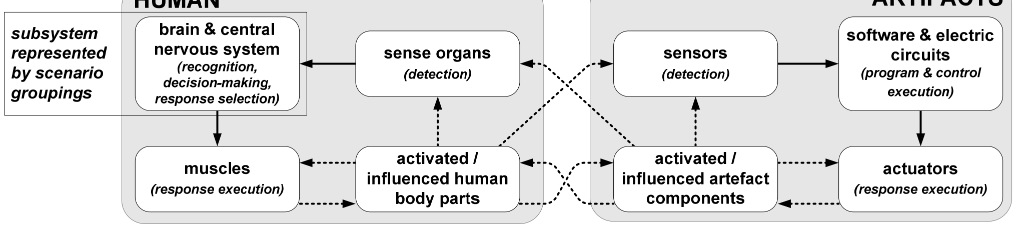

HUMAN
ARTIFACTS
subsystem represented by scenario groupings
brain & central nervous system
(recognition, decision-making, response selection)
sense organs
(detection)
sensors
(detection)
software & electric circuits
(program & control execution)
muscles
(response execution)
activated / influenced human body parts
activated / influenced artefact components
actuators
(response execution)

Figure 4. *Combining the control and interaction behaviors as found in existing scenario-based simulation approaches. Legend: see Figure 2.*

tively. It has been shown that including probability is possible with minor adaptations to the original statecharts [31]. To include timing, we will use the opportunities offered outside the scenario bundle by other modeling elements involved in the co-simulation.

Focusing on human control of interaction, we have established that the scenario bundle is a statechart that represents and, during simulation, executes human logical behavior. As one of the partial models involved in a co-simulation, it represents key aspects of human control. Aspects of perception on the input side, and of force exertion the output side of the chain of human control, however, are represented in other partial models. As was mentioned above, timing in human logical behavior is also modeled and simulated outside the statechart.

In section 4, starting with the scenario-bundle as a 'black box' with inputs and outputs typical for statecharts in simulation, we will elaborate the integral simulation representation of the human-artifact system in which it is embedded. For the representation and simulation of force exertion based on proportional control we refer to [24]. In this previous work, we have focused on mechanical interactions, leaving open other sorts of physical interaction (thermal, acoustic, etc.) for future elaboration. For now, we continue to assume that physical human-artifact interactions are mechanical. In completing the integral simulation model, we will also elaborate our proposed representation of human perception and a solution for modeling and simulation of artifact control behavior, i.e., the top right-hand side of Figure 4, which was not discussed so far. After that, we will explain the contents of our control models in detail, with emphasis on the scenario bundle in section 5.

4. INTEGRAL SIMULATION REPRESENTATION

Our starting point in the development of the integral model has been a scenario bundle modeled as a statechart. In simulation, this model sends *control signals* to the physical interaction model, prescribing movements to muscles. Decision-making in the scenario bundle operates on outputs of the physical simulation, which have been pre-processed by a model of human perception. The simulation outputs of the physical interaction model consist of continuous data streams that represent values of user-defined variables in the physically based interaction model. These data streams are called *meter signals*.

4.1. Outputs of the scenario bundle

The control signals produced by scenario bundle contain information about (angular) velocities and/or positions (or angles, respectively) that the physics simulation uses to compute muscle forces. During simulation, each signal continuously has the latest required value for each degree of freedom of the controlled limbs. Output signals are described as functions of time. Changes in the description of an output signal are caused by the logical simulation. In the simplest case, an output signal is a constant, and a change implies a transition to a different constant value (for instance, to prescribe a change in a constant velocity).

A problem of statecharts is that, internally, they offer no means to model timing. For instance, it is possible to model a transition that waits for a given external stimulus (event), but it is not possible to modeling human latency as 'wait for a duration *t*', with *t* a given time interval. The issue is resolved by letting the statechart produce an outgoing event each time a waiting interval starts, together with an outgoing value corresponding to the duration of the interval. This outgoing *start-delay event* is processed by starting a timer in a timing sub-model outside the statechart, which produces the necessary external *end-delay event* that is processed by the statechart to end the delay.

4.2. Inputs of the scenario bundle

The inputs on which a statechart reacts in simulation must be specified as events. The problem is that to obtain these events, some simulation of human perception is needed to convert the continuous data streams produced by the physics model. We resolved this by a applying a simplified interpretation of perception, in which only those human observations are modeled that are needed for decision-making. With 'observation', we denote the detection of the signal *and* the recognition of that particular change in the signal that triggers the actual instantiation of a particular event. If the observation is instantaneous input for decision-making, a modeled *recognition event* signifies that the value of a given meter signal, or generally of a function of multiple meter signals, crosses a specified threshold. If the observation is not for instantaneous decision-making, but it represents a value that is needed to evaluate conditions associated with decisions at arbitrary times, the observation is modeled to be sent to the scenario bundle as *recognition data*. Our simplification of perception presupposes that the simulated human detects and recognizes every signal that is relevant for execution of the scenario-bundle simulation. In other words, we ignore the possibility of detection and recognition errors in simulation.

Two additional special types of events that are needed as input to the scenario bundle are *end-delay events* as mentioned in 4.1 and the *start event*, which is needed to activate the statechart when the user starts a simulation.

Table 1 gives an overview of the interfaces described in 4.1 and 4.2.

Table 1 *Overview of control behaviors, partial models, representations and signal connections*

control behavior		*partial model*	*representation*	*input signals*	*output signals*
detection	}'observations'	input interface & timing model	block diagram	data from physics model	events to scenario bundle
recognition					
latency				events and data from scenario bundle	
decision-making		scenario bundle	statechart	events from input interface model	data to physics model
response selection					
response execution	(in terms of velocities and positions)				
	(in terms of forces)	physics model	equation	data from scenario bundle	data to input interface model

4.3. Integrating the simulation elements

To complete the integral simulation model, we must include a partial model that can simulate information processing by artifacts. Since the statecharts we selected for modeling and simulation of human information processing originate from the need to model information processing in artifacts [28], and since they have also been employed for simulation purposes in that area (e.g., [32]), we have decided to use the same representation as we are using for human information processing. Statecharts of artifact information-processing are called *procedure structures*, since they represent the procedures to process information streams in artifacts. A procedure structure is connected to the other simulation models in the same way as a scenario bundle. Since both the scenario bundle and the procedure structure communicate with the same physics simulation, we use the input and timing interface to connect both to the meter signals.

Figure 5 is a block-diagram representation of the general signal flows in the complete model during co-simulation of the logical models (procedure structure and scenario bundle), the input and timing interface model, and the physics model. The diagram, that corresponds to Figure 4 rotated 90° to the left, shows that generally n scenario bundles and m procedure structures can be connected to the input and timing interface model, which corresponds to a system of n virtual humans interacting with m virtual artifacts, with $n,m \geq 0$.

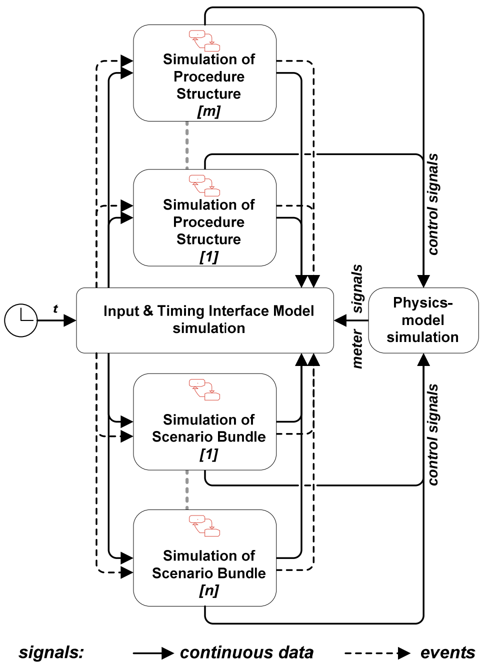

Simulation of Procedure Structure [m]
Simulation of Procedure Structure [1]
t
Input & Timing Interface Model simulation
meter signals
Physics-model simulation
control signals
control signals
Simulation of Scenario Bundle [1]
Simulation of Scenario Bundle [n]
signals:
continuous data
events

Figure 5. *Signal flows in human-artifact interaction simulation*

5. CONTROL MODELS

5.1. The scenario bundle: a logical model of human control

Both the procedure structure and the scenario bundle are modeled as statecharts. The elaboration below focuses on the scenario bundle, since it is the main topic of this paper.

To represent logical models we used the Simulink Stateflow notation, a dialect of statecharts [33]. Figure 6 shows an example of the graphical appearance of a statechart in Stateflow. In 5.1.1-5.1.3, we elaborate on the specific modeling entities that are used in the scenario bundle, namely, input and output ports, events, states, transitions (with conditions and actions), and data. In 5.1.4 we explain how the distinction between the three levels of human logical control (decision-making, response selection, and response execution) is maintained in the scenario bundle.

5.1.1. Input and output ports

A scenario bundle is externally connected by four types of input and output ports, namely, ports for incoming events and outgoing events, as well as ports for incoming data and outgoing data. All events appear as pulse signals and all data appear as continuous streams. Each incoming or outgoing event is assigned to a port, which specifies the name of the event, and whether it is incoming or outgoing. Likewise, each incoming our outgoing data stream has a port that specifies its signal name, its data type, and whether it is incoming or outgoing. Apart from these external data streams, internal data may be

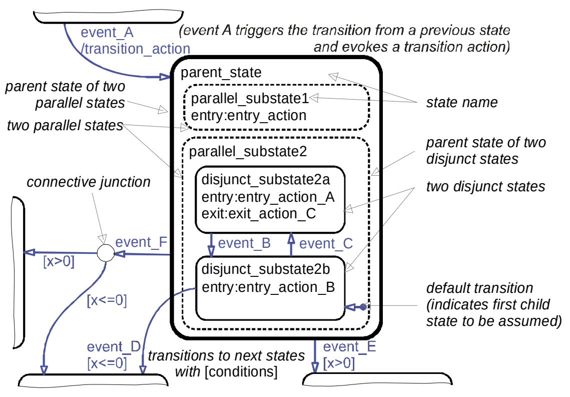


Figure 6. *Excerpt from an example Stateflow diagram with its key graphical modeling entities*

defined that is used within the scenario bundle only and therefore not associated to a port. Figure 7 shows the classification of signals that scenario bundles process during simulation and which are further explained below.

5.1.2. Events, states, transitions, conditions, and actions

Being a statechart, a scenario bundle is a finite state machine that allows concurrency, which means that it describes a system that is always in at least one of a finite set of states. Triggered by *external* or *internal events* it performs transitions (represented by arrows) between states (represented as rounded rectangles). External events are pulse signals that the scenario bundle receives as input. Apart from the start event, they are either recognition events or end-delay events that have been generated by the input and timing interface. Internal events are signals that a scenario bundle generates based on changes within itself. They are typically used to synchronize transitions inside parallel states.

A transition may be associated to a specified event, and logical [conditions] may be added, so that it is only taken if the specified event occurs *and* the [conditions] apply. Additionally, /actions associated to transitions and states in the scenario bundle can be defined to generate internal or outgoing external events or changes in data values.

5.1.3. Data

In the scenario bundle we distinguish external incoming data, external outgoing data, and internal data. All external incoming data comes from recognition-data signals, i.e., values originating from the physics model or from other logical models, which appear as variables in [conditions] attached to state transitions. For the external outgoing data, we distinguish control signals for physics simulation, timing durations for the input and timing interface, and data intended for other logical models as recognition data. Changes in outgoing external data signals appear as the result of /actions attached to transitions or states. Finally, internal data signals are used if information processing produces intermediate data

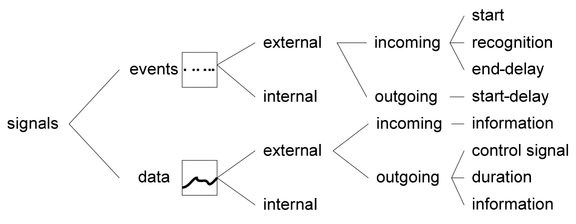


Figure 7. *Classification of the signals processed by a scenario bundle*

that is used within the logical simulation of the scenario bundle itself.

5.1.4. Levels of human control

To represent the three levels of human control, the scenario bundle is hierarchically structured into layers: a decision-making layer, a response selection layer, and a response execution layer (Figure 8), the latter two consisting of sub-charts of statecharts at higher levels. A special group of sub-charts is the group of *response-execution primitives*, which are specified as a part of the response execution layer. The primitives contain basic low-level control commands for the movement of each limb in one of its degrees of freedom (e.g., move forward, rest, and move backward). Because the simulation should be able to call these commands from any state or transition in the scenario bundle, a parent state of these commands is included at the highest level, parallel to the decision-making statechart.

To facilitate organization of the graphical representations, each of the three layers can be decomposed into an arbitrary number of sub-layers. As an example, Figure 8 shows one layer for decision-making, two sub-layers for response selection, and, apart from the response execution primitives, one additional sub-layer for response execution.

The layered representation of scenario bundles has been derived from common theories on human motor control. This means that unlike the descriptions in the previous subsections it does not apply to artifacts and therefore not to procedure structures.

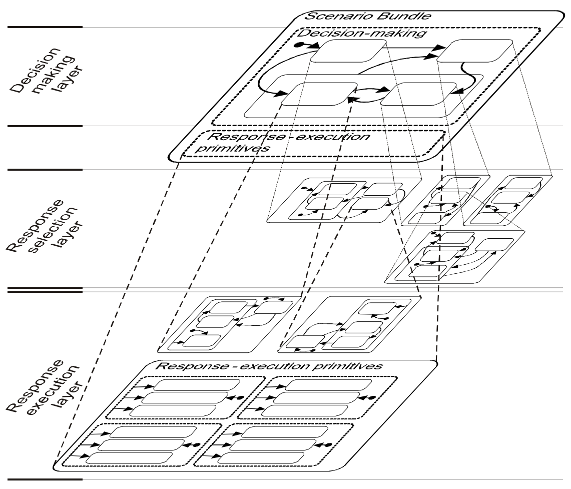


Figure 8. *Layers of the scenario bundle*

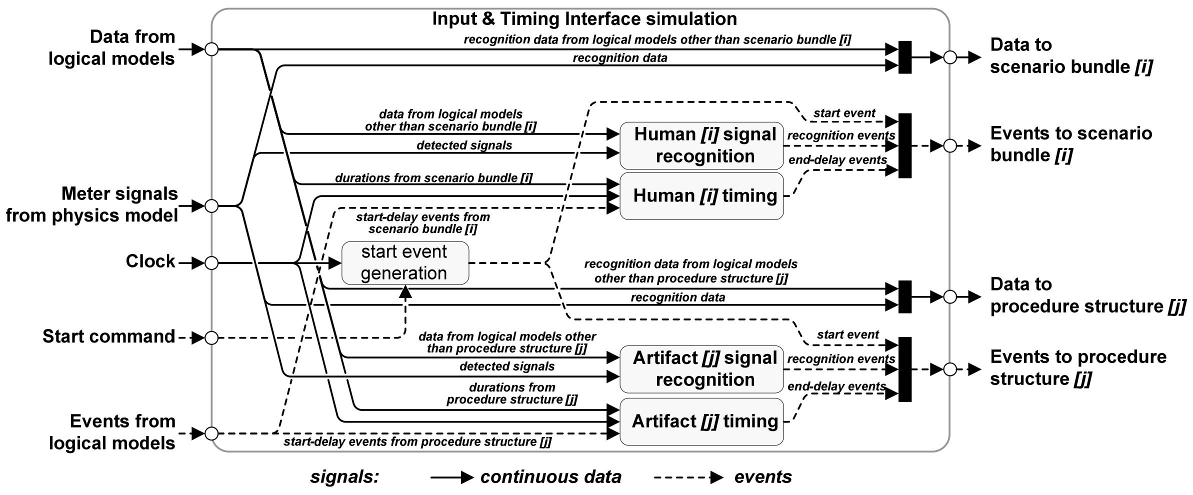

Input & Timing Interface simulation
Data from logical models
Meter signals from physics model
Clock
Start command
Events from logical models
recognition data from logical models other than scenario bundle [i]
recognition data
data from logical models other than scenario bundle [i]
detected signals
durations from scenario bundle [i]
start-delay events from scenario bundle [i]
start event generation
Human [i] signal recognition
Human [i] timing
start event
recognition events
end-delay events
recognition data from logical models other than procedure structure [j]
recognition data
data from logical models other than procedure structure [j]
detected signals
durations from procedure structure [j]
start-delay events from procedure structure [j]
Artifact [j] signal recognition
Artifact [j] timing
start event
recognition events
end-delay events
Data to scenario bundle [i]
Events to scenario bundle [i]
Data to procedure structure [j]
Events to procedure structure [j]
signals:
continuous data
events

Figure 9. *Signal flows of the input and timing interface*

5.2. Input and timing interface model

Figure 9 shows the signal flows of the input and timing interface model for the typical case that one scenario bundle and one procedure structure are connected to a physics model. It performs the following operations on signals:

– Passing on recognition data from one model to another without further processing. The source can be either the physics model or a logical model; the destination is always a logical model.
– Generating a start event to start logical simulations. This event is generated by a user command and sent to all logical models. At the same time, simulation time starts running.
– Generating recognition events from meter signals that originate from the physics model. A recognition event signifies that a given meter signal produced by the physics model crosses a specified threshold value. Depending on whether the threshold value is crossed while increasing or decreasing, recognition events are either based on *rising-edge, falling-edge, or either-edge* triggers.
– Generating end-delay events. An end-delay event is based on a start-delay event e_{sd}, a given duration $\Delta\, t_d$ (data from the logical model in which the delay is to take place), and the simulation time t, so that e_{ed} is sent to the logical model at $t(e_{ed}) = t(e_{sd}) + \Delta\, t_d$.

5.3. Physics model

For the control of simulations, it is only important that the physics model can calculate muscle forces from control signals representing (angular) velocities, displacements, and angles. It is not important how the physics model is built and on which principles it has been based. It can be a 3D model, a 2D (schematic) model, or even a numerical algorithm that represents a set of differential equations describing the physical behavior of a human-artifact system. It is only required that (i) it represents the whole human-artifact system and (ii) that it accepts control signals and produces meter signals that represent all the necessary parameters for the simulation of decision-making and other information processing in human-artifact interaction.

6. DEVELOPMENT OF A PROOF-OF-CONCEPT IMPLEMENTATION

In testing our simulation method, our priority has been to verify the simulation capabilities in terms of the combined functionality that existing methods do not offer. Because of the limited scope of our testing objectives, our priority has not been to develop dedicated user interfaces for the creation of the various models and their connections. Instead, we relied on available commercial software packages that offer such interfaces without compromising the contents of the models according to our specifications in sections 3-5.
We used MATLAB/Simulink Stateflow to model and simulate scenario bundles and procedure structures, MSC ADAMS to model and simulate the physics (mechanics) of human-artifact-interactions, and MATLAB/Simulink to connect these models.
The objectives of testing were to evaluate:

– the feasibility of simulating a representative selection of basic physical human-artifact interactions, such as reaching, grasping, and manipulating the position of an object. However, for the proof of concept implementation, we did not strive for realization of accurately simulated natural human motion patterns;
– the feasibility of simulating varying courses of the use process, i.e., different paths through a scenario bundle;
– simulation of scenario bundles with variations of the product design
– simulation performance;

Figure 10 shows the snack dispenser that was modeled and simulated as a test case for the proof-of-concept implementation. After a customer has

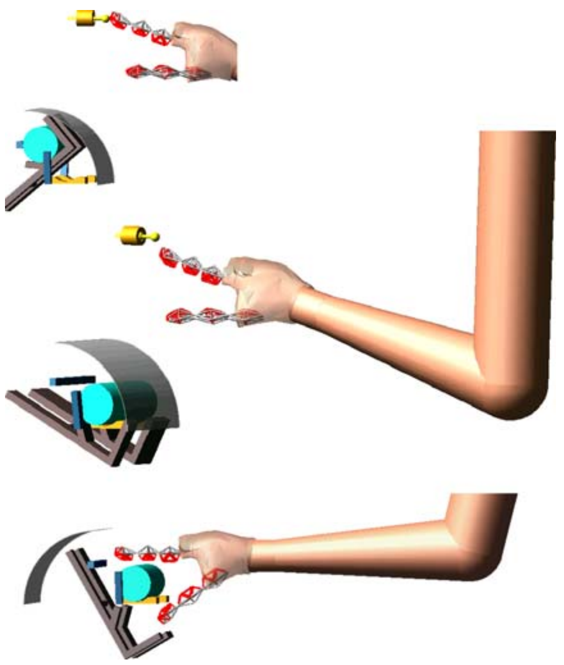


Figure 10. *A snack dispenser – the human-artifact system that was modeled and simulated with MSC ADAMS and MATLAB/Simulink as a case study.*

pressed a button, a snack is delivered. The dispenser has some built-in logic that controls the release of the snack to the customer. The snack is supposed to be kept cold, and if the customer does not grab it within a given interval after pushing the button, it is put back into storage. One of the alternative scenarios we could include because of this behavior is a retry loop to be carried out by the user if this happens.

The picture shows the physics model, which was created with MSC ADAMS. Since our test case merely serves as a proof of ideas, the model – especially the part that represents the human – has been simplified to reduce the computation time and the effort required for modeling. We only modeled an arm and a hand with only a thumb and an index finger. The fingers have been modeled as low-resolution particle systems to make simulation possible of the large deformations that are needed for a firm grip on objects when grasping. Relevant for modeling human motor control is that we simplified the actuation of limbs by providing each limb with only one muscle, which performs both contraction and extension. Other simplifications are discussed in [24].

To begin with, the remainder of this section provides a detailed description of the scenario bundle that describes the control of human-product interaction in 6.1. Then we will present the procedure structure and the input and timing interface model in 6.2. After that, in section 7, we will elaborate on testing the proof of concept implementation by running simula-

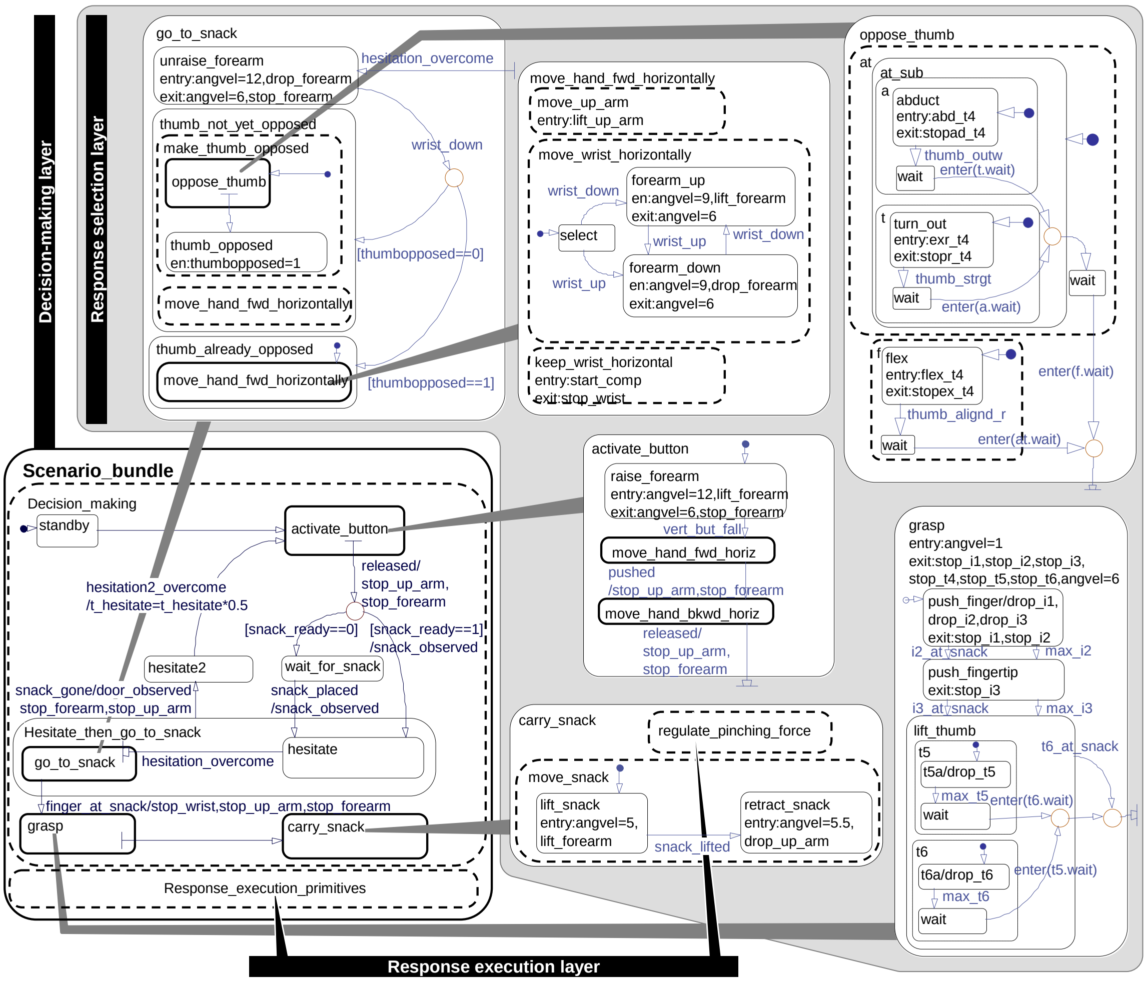


Figure 11. *Decision-making and response-selection layers of the scenario-bundle of the use of the snack dispenser. The response-execution layer is shown in Figure 12.*

tions.

6.1. Scenario bundle of the snack dispenser

The simulated system involves one human, and one information-processing artifact. Therefore, our logical models are one scenario bundle for the customer and one procedure structure for the snack dispenser. These were modeled using Simulink Stateflow R2007a. Figure 11 shows the decision-making layer and the response-selection layer of the scenario structure. The response selection layer can be arbitrarily decomposed by the user, but we applied this particular decomposition because particular lower-level decision-making processes concern basic motion and manipulation patterns that are likely to be involved in many use processes of many products. Making these available as distinct modeling entities ensures easy availability for reuse in other projects. Typical examples of such lower-level decision-making patterns are ‘oppose thumb’ and ‘activate button’.

Each scenario starts with activating the button after the start event has triggered the outgoing transition of the initial stand-by state. Once the button has been released, the virtual customer moves his hand towards the snack and takes it. To include human latency in the model, a state hesitate has been included. If the virtual customer hesitates for too long, the procedure structure (see Figure 13) issues a command to put the snack back into storage space. By using a random generator for the value t_hesitate we could also introduce a probability aspect.

Once the hand is positioned close enough to the snack, a proximity sensor in the snack dispenser sends an event within the procedure structure, which prevents the snack from being put back. If, as a result of hesitation, the snack has been taken away and the door has been closed (snack_gone), the customer tries again. This retry loop includes another hesitation (the human is surprised to see the snack disappearing), and the next time the human realizes that he should not hesitate so long after having pushed the button (/t_hesitate = t_hesitate*0.5). When the hand can reach the snack, it is grasped (grasp) and taken (remove_snack).

Figure 12 shows part of the response execution layer, which contains a routine regulate_pinching_force for holding the snack while carrying, and the response execution primitives. There is a response execution primitive for each degree of freedom of each limb, eleven in total: for each, a parent states contains a set of disjunct child states. The parent states are parallel to each other and parallel to all the other states in the scenario bundle (Figure 11). This means that any child state can be activated at any time, but for each limb, only one child state is active at a time. For one degree of freedom of each limb, there are typically three basic child states, two for moving up and down (or inward and outward), and one for resting, which is the default state. The forearm, the upper arm, the wrist, the phalanges of the index finger (i1, i2, i3) and the middle and distal phalange of the thumb (t5, t6) each have two degrees of freedom; the proxal phalanx of the thumb (th4) has three degrees of freedom, so that the thumb can be opposed.

To (de)activate motion of limbs, internal events are used, which are to be included as actions in one of the higher-level diagrams in Figure 11. For instance, wherever in Figure 11 an action contains the command lift_forearm, the child state forearm.up is assumed and a control signal for the angular velocity of the forearm is set to a positive value equal to 0.1*angvel, with angvel a constant that has been defined as internal data of the scenario bundle.

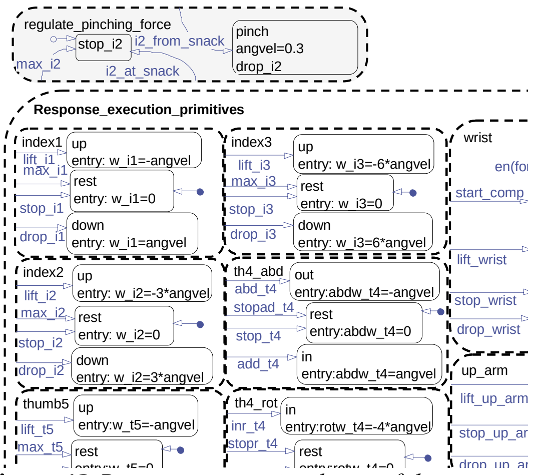


Figure 12 *Response-execution layer of the scenario bundle of the use of the snack dispenser (detail).*

6.2. Procedure structure and input and timing interface model

Figure 13 shows the procedure structure of the snack dispenser. The internal programming of the product was kept simple: additional functionality of delivering subsequent snacks was not included in our model, as was not the usual functionality to collect payments.

Using Simulink, the logical models have been connected to the physics model according to Figure 5. The input and timing interface model is a Simulink

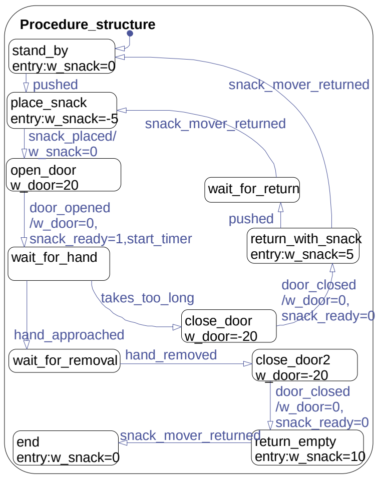


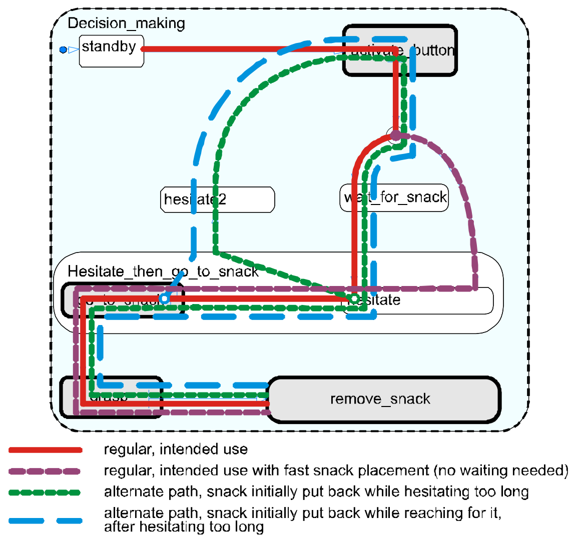
Figure 13. *Procedure structure of the snack dispenser*

subsystem that (i) converts continuous data streams containing meter values to events using *hit crossing* block-diagram elements and (ii) generates end-delay events by using *sample and hold* blocks that react on start-delay events by counting down from the given duration until the threshold value zero has been reached.

7. RUNNING SIMULATIONS: RESULTS AND DISCUSSION

During and after co-simulation, the commercial software that we used provides feedback to the user as follows. The course through the scenario bundle can be followed instantly, because Stateflow animates the logical models by highlighting states and transitions when active. The simulation of mechanical behavior in ADAMS can also be animated during the simulation, but this considerably reduces computational performance. Therefore we chose to run ADAMS in ‘batch mode’, addressing its solver algorithm only. The 3D animation can be viewed afterwards. To check progress of physical interaction during the simulation, XY plots of the position of key points on the human body and the snack are produced by processing meter signals in Simulink.
We evaluated the simulations based on (i) the feasibility of simulating a representative selection of basic physical human-artifact interactions, (ii) the possibility of simulating scenario bundles with variations of the product design, (iii) the possibility to simulate multiple scenarios from one bundle, and (iv) simulation performance.

Basic interactions. We were able to simulate the following basic interactions that we included in the scenario structure: (i) reaching, (ii) pushing and (iii) releasing a button, (iv) grasping, and (v) carrying an object. To keep the control of limbs in the proof-of-concept implementation simple, we modeled most of the hand motions based on ‘joint-space planning’ rather than on ‘hand-space planning’. This means that the scenario bundle prescribes rotations of joints rather than motion paths of the hand [10]. Although humans are capable of exerting this form of control, it results in somewhat awkward ‘robotic’ and indirect motion patterns, whereas hand-space planning would result in sophisticated motions straight towards the target. We included some basic hand-space planning in straight horizontal motions of the hand, but did not implement it to move the hand to arbitrary targets.
Control of grasping was modeled by stopping forward motion of the hand if a given horizontal overlap between the index finger and the snack has been reached. Then, first the two distal phalanges of the index finger are lowered until they touch the snack. After that, the thumb is raised until it touches the snack, and then the hand with snack is lifted. It turned out that the physics simulation is very susceptible to small changes in the thresholds of the events



Decision_making
standby
activate_button
hesitate2
wait_for_snack
Hesitate_then_go_to_snack
go_to_snack
hesitate
grasp
remove_snack
regular, intended use
regular, intended use with fast snack placement (no waiting needed)
alternate path, snack initially put back while hesitating too long
alternate path, snack initially put back while reaching for it, after hesitating too long

Figure 14. *Simulated variety of paths through the scenario bundle (cf. Figure 11)*

involved, which cause the simulation to crash, or to result in the snack falling out of the hand when lifting. The threshold values that we successfully used in the final simulations were established by trial and error. We expect that a more realistic model of the hand with five fingers and with high-resolution particle clouds will make simulation of a firmer grip possible and thus resolve this issue.

Simulating multiple scenarios from one bundle
Figure 14 shows four of the different paths through the scenario bundle, which we were operationalized at ‘playing’ with the preset values of human latency (hesitation time), and with the angular velocities imposed on actuators in the snack dispenser. Other scenarios with multiple retries could also be simulated (not in the figure). The four paths in Figure 14 correspond to three different paths in the procedure structure.

Design variations. The models that we used to obtain our proof of the concept were built from starting with simple versions of the snack dispenser, the scenario bundle, and the procedure structure. These were gradually improved and made more complex by using feedback from early simulation runs, in which we observed unanticipated behaviors. These outcomes prompted us to adapt the physical appearance of the product as well as the program structure, which is also part of the product design. The role of simulation feedback in this workflow is similar to what we expect in support of our intended end users, i.e., product designers.
With the ‘final’ versions of the models presented in this paper it is possible to investigate small variations that do not require definition of new meter signals and control signals. We confirmed this by varying the design of the physical appearance by changing the location of the button. When raising or lowering the position by 25mm, the same scenario bundle could still control the simulation.

Simulation performance. The simulation time needed for a use process with duration of 5.4s, corresponding to the fourth path in Figure 14, was 36:53 minutes on a PC with Intel Core 2 Quad 2.66 GHz CPU running Windows XP SP2, using three threads for the ADAMS 2007 simulation. During the simulation, ADAMS used 20-30% of CPU time, while Simulink used 3-5%. On average, this simulation took almost 7 minutes of computation for each second of simulated behavior. Thus, in the current proof-of-concept implementation, real-time simulation is not feasible, and the physics simulation appears to be the bottle-neck. In a non-interactive simulation setup, real-time performance is typically of minor importance because the simulated system does not need to be continuously synchronized with a real system. According to [34], the acceptable simulation time for practical engineering solutions is somewhere below ten hours. Our current proof-of-concept implementation appears to be well within that limit.

8. CONCLUSIONS AND FUTURE WORK

In this paper the concept of controlling simulations of human-artifact interaction with scenario bundles has been elaborated to such a level that it could be tested in an application case study. We found that our approach could be successfully used to (i) simulate a representative selection of basic physical human-artifact interactions (ii) simulate multiple scenarios from one bundle, and (iii) simulate variations of the product design to obtain feedback about the product design. From phenomenological evidence (i.e., from our own test case), we found that the latter applies both to small variations in an established design as to variations that were implemented as part of improvement cycles during design. While this is no scientific proof, it is a first indication of how our modeling and simulation approach may be able to help designers during conceptual design.

The simulation performance appears to be acceptable, but we found that here, simulation of the scenario bundle is not the bottle-neck anyway.

Regarding scenario bundles, we identified two main issues that need to be addressed in future work. Firstly, more detail knowledge about human motion patterns, for instance about hand-space motion planning and about control of grasping, has to be included in scenario bundles. Preferably, this knowledge should be made available as modules that can easily be inserted in scenario bundles that describe the use of different products.

Secondly, the software and the interfaces used to prepare and connect the simulation models have not been developed from the aspect of being used by product designers. The logical models have been created using the graphical interface of Stateflow, and connections have been manually assigned by assigning variables to ADAMS output signals, and by manually creating the input and timing interface model as a Simulink block diagram. We expect that these tasks can also be supported by a certain level of automation. Additional research is needed to assess how much product designers can familiarize themselves with statecharts, or with alternative ways of building logical models.

References

[1] Carroll, J.M., Mack, R.L., Robertson, S.P., Rosson, M.B., 1994, Binding objects to scenarios of use, Int. J. Human-Computer studies, Vol. 41, pp. 243-279

[2] Hooper, J., Hsia, P., 1982, Scenario-based prototyping for requirements identification, Software Engineering Notes, Vol. 7, No.5, pp. 88-93

[3] Jarke, M., Tung Bui, X., Carroll, J.M., 1998, Scenario management: an interdisciplinary approach, Requirements Engineering, No. 3, pp.155-173

[4] Welker, K., Sanders, E.B.N., Couch, J.S., 1997, Design scenarios - to understand the user, Innovation, Fall 1997, pp. 24-27

[5] Cremer, J., Kearney, J., Papelis, Y., 1995, HCSM: a framework for behavior and scenario control in virual environments, ACM Transactions on Modeling and Computer Simulation, Vol. 5, No.3, pp 242-267

[6] Honglun, H., Shouquian, S., Yunhe, P., 2007, Research on virtual human in ergonomic simulation, Computers & Industrial Engineering, Vol. 53, pp. 350-356

[7] Newell, A., Simon, H.A., 1971, Simulation of human thought, In: Dutton, J.M., Starbuck, W.H. (eds.), Computer simulation of human behavior. New York: John Wiley and Sons, Inc.

[8] Stanton, N.A., Baber, C., 1998, A systems analysis of consumer products, In: Stanton, N.A.(Ed.), Human factors in consumer products, London: Taylor & Francis Ltd., pp. 75-90

[9] Hsia, P., Samuel, J., Gao, J., Kung,D., Toyoshima, Y., Chen, C., 1994, Formal approach to scenario analysis, IEEE Software, March 1994, pp. 33-41

[10] Rosenbaum, D.A., 1991, Human motor control, San Diego: Academic press

[11] Lindsay, P.H., Norman, D.A., 1972, Human information processing - an introduction to psychology, New York:Academic Press

[12] Eysenck, M.W., Keane, M.T., 2000, Cognitive Psychology (4th Edition), Hove, UK: Psychology Press

[13] Powell, A., Katzko, M., Royce, J.R., 1978, A multi-systems theory of the structure and dynamics of motor functions, Journal of Motor Behavior, Vol. 10, No. 3, pp. 191-210

[14] Stelmach, G.E., 1982, Information-processing framework for understanding human motor behavior, In: Kelso, J.A.S.

(Ed): Human motor behavior: an introduction, pp 63-91. London: Lawrence Erlbaum Associates

[15] Jagacisnki, R.J., Flach, J.M., 2003, Control theory for humans, Mahwah (NJ): Lawrence Erlbaum Associates

[16] Costello, R.G., 1968, The surge model of the well-trained human operator in simple manual control, IEEE Transactions on Man-Machine Systems, Vol. 9, No. 1, pp. 2-9

[17] Abend, W., Bizzi, E., Morasso, P., 1982, Human arm trajectory formation, Brain, Vol. 105, pp. 331-348

[18] Mutha, P.K., Sainburg, R.L., 2007, Control of velocity and position in single joint movements, Human Movement Science, Vol. 26, pp. 808-823

[19] Rauterberg, M., Fjeld, M., Schluep, S., 1997, Goal setting mechanism in Petri net models of human decision making, Proceedings of the IEEE Int. Conference on Systems, Man and Cybernetics, Orlando FL, pp. 2696-2701

[20] Filla, R., 2005, An event-driven operator model for dynamic simulation of construction machinery, Proceedings of the 9th Scandinavian Conference on Fluid Power, Linköping, Sweden

[21] Zeigler, B.P., Praehofer, H., Kim, T.G., 2000, Theory of modeling and simulation - integrating discrete event and continuous complex dynamic systems, second edition., San Diego: Academic press

[22] Korn, G.A., Wait, J.V., 1978, Digital continuous system simlation, Englewood Cliffs: Prentice-Hall, Inc.

[23] Van der Vegte, W.F., 2006, A survey of artifact-simulation approaches from the perspective of application to use processes of consumer durables, Proceedings of the TMCE, Ljubljana, pp. 617-632

[24] Van der Vegte, W.F., Horváth, I., Rusák, Z., 2008, Simulating the use of products: applying the nucleus paradigm to resource-integrated virtual interaction models, Proceedings of TMCE, Izmir, pp. 591-605

[25] Anderson, J.R., Bothell, D., Byrne, M.D., Lebiere, C., 2004, An integrated theory of the mind, Psychological Review, Vol. 111, No. 4, pp. 1036-1060

[26] Gill, A., 1962, Introduction to the theory of finite-state machines, New York: McGraw-Hill book company, Inc.

[27] MacDonald, I.L., Zucchini, W., 1997, Hidden Markov and other models for discrete-valued time series, Lonndon: Chapman & Hall

[28] Harel, D., 1987, Statecharts: a visual formalism for complex systems, Science of Computer Programming, Vol. 8, pp. 231-274

[29] Jahanian, F., Mok, A.K., 1994, Modechart: a specification language for real-time systems, IEEE Transactions on Doftware Engineering, Vol. 20, No. 12, pp. 933-941

[30] Petri, C.A., 1962, Fundamentals of a theory of asynchronous information flow, Proceedings of the 1962 IFIP Congress, Amsterdam, pp. 386-390

[31] Francês, C.R.L., Da Luz Oliveira, Costa, J.C.W.A., Santana, M.J., Santana, R.H.C., Bruschi, S.M., Vijaykumar, N.L., De Carvalho, S.V., 2005, Performance evaluation based on system modeling using Statecharts extensions, Simulation Modelling Practice and Theory, Vol. 13, pp. 584–618

[32] Kendall, I.R., Jones, R.P., 1999, An investigation into the use of hardware-in-the-loop simulation testing for automotive electronic control systems, Control Engineering Practice, Vol. 7, pp. 1343-1356

[33] MathWorks, Inc., the, 2007, Stateflow and stateflow coder for use with Simulink - modeling, simulation, implementation, Natick: The MathWorks, Inc.

[34] Dreisbach, R.L., 2008, Digital simulation and computational technologies in the aerospace industry, proceedings of TMCE, Izmir, pp. 19-24. *Note: the number we referred to was shown in the presentation; it is not in the paper.*