

Adding animated presentation agents to the interface

Thomas Rist, Elisabeth André, Jochen Müller

Angaben zur Veröffentlichung / Publication details:

Rist, Thomas, Elisabeth André, and Jochen Müller. 1997. "Adding animated presentation agents to the interface." In Proceedings of the 2nd international conference on Intelligent user interfaces - IUI '97, Orlando, Florida, USA, January 06 - 09, 1997, edited by Johanna Moore, Ernest Edmonds, and Angel Puerta, 79-86. New York, NY: ACM Press. <https://doi.org/10.1145/238218.238298>.



Adding Animated Presentation Agents to the Interface

Thomas Rist, Elisabeth André, Jochen Müller
German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
Email: {rist, andre, jmueller}@dfki.uni-sb.de

ABSTRACT

A growing number of research projects both in academia and industries have started to investigate the use of animated agents in the interface. Such agents, either based on real video, cartoon-style drawings or even model-based 3D graphics, are likely to become integral parts of future user interfaces. To be useful, however, interface agents have to be intelligent in the sense that they exhibit a reasonable behavior. In this paper, we present a system that uses a lifelike character, the so-called PPP Persona, to present multimedia material to the user. This material has been either automatically generated or fetched from the web and modified if necessary. The underlying approach is based on our previous work on multimedia presentation planning. This core approach is complemented by additional concepts, namely the temporal coordination of presentation acts and the consideration of the human-factors dimension of the added visual metaphor.

Keywords

Animated user interface agents, presentation techniques for web applications, automated multimedia authoring

INTRODUCTION

Through the last decade, the design of user interfaces has undergone dramatic changes. The introduction of multiwindowing and the exploitation of multimedia technology that allows for presentations which incorporate written and spoken text, sound, graphics, animation, and even virtual 3D reality has opened up completely new possibilities. The next major step in the evolution of interfaces is very likely to focus on highly personalized interfaces. Personalization refers to the ability of a system to design multimedia presentations on the fly in order to meet the information needs of individual users in particular situations. This means that decisions on content selection, media choice, and medium-specific encoding of information are made with regard to models of the user, task, and situation.

While this kind of personalization is widely addressed in the area of user modelling and in research on intelligent presentation systems, less attention has been paid to the *peripheral* aspect of personalizing user interfaces so far. By peripheral personalization we mean that the system personifies itself audio-visually, e.g. as in our case, by means of an animated character called PPP Persona. There are several motivations for using such an animated presentation agent in the interface. First of all, it adds expressive power to a system's presentation skills. For example, cross-references between different media (possibly occurring in different windows) can be effectively established through a two-handed pointing gesture. If one strives for emulating the multimodal interaction that occurs between humans, the presentation agent's repertoire of behaviors may even comprise mimics to express emotions. Furthermore, a presentation agent can also serve as a guide through a presentation to release the user from orientation and navigation problems known from multiwindow/multi-screen settings. Last but not least, there is the entertaining and emotional function of such an animated character. It may help to lower the "getting started barrier" for novice users of computers/applications, and, as Adelson notes, "... interface agents can be valuable educational aids since they can engage students without distracting or distancing them from the learning experience" (cf. [1], pp. 355).

Our interest in animated presentation agents arose from our previous work on the development of the knowledge-based presentation system WIP (cf. [27]). Although the presentations (texts, pictures, animations, and mixed presentations) synthesized by WIP are coherent and tailored to the individual settings of given presentation parameters (target language, user characteristics, document type, and resource limitations), WIP does not plan when and how to present the generated material to the user. To enhance the effectiveness of presentations, we aimed at an augmented system (called PPP) in which an animated character plays the role of a presenter, showing, explaining, and verbally commenting textual and graphical output on a window-based interface. Some of our current sample applications are illustrated in the next subsections. After that, we sketch the approach for planning dynamic presentations and provide an overview of the PPP system and a server component that coordinates the behavior of the PPP Persona. The paper ends with a com-

parison with related work and some concluding remarks.

Automatically Generated Instructions

The first application scenario deals with instructions for the maintenance, and repair of technical devices, such as modems. In the ideal case, instructions are customized to the individual needs of a particular user. Doing this manually, however, is not feasible for many applications. It would require to prepare an exponential number of presentations in advance and to hold them on stock. In contrast to the labor-intensive and thus costly manual authoring of instructions, the PPP system generates on the fly both all the material to be presented (such as textual explanations, illustrations, and animation sequences) and a plan for presenting the material in a temporally coordinated manner.

For illustration, let's consider the task of explaining how to operate a modem. That is, we start the PPP system with the presentation task: *Explain modem*. Starting from this high-level task, the system produces an audio visual presentation given by the interface agent. The screen shots shown in Fig. 1 to Fig. 3 were taken from this presentation. At the beginning, a window was created that contains a depiction of the modem's circuit board. After the window has appeared on the screen, the PPP Persona verbally informs the user that there is a socket for the phonenumber cable. Since several modem parts are visible from the chosen viewing angle, the system decided to resolve the ambiguity by means of a labeling act. While a conventional graphics display would rely on text labels, the PPP Persona enables the realization of dynamic annotation forms as well. In the example, it points to the socket and utters "This is the phonenumber socket" (using a speech synthesizer). One advantage of this method over static annotations is that the system can influence the temporal order in which the user processes a graphical depiction. It is even possible to combine both methods since the PPP Persona can also place textual labels on the graphics before the user's eyes. After that, the Persona describes the remaining actions to be carried out. Finally, the user is told that a certain LED will light up after the modem has been turned on. Again, the PPP Persona points to the respective object and utters "This is the LED 11". However, since the employed annotation strategy aims at bringing the object to which the Persona should point into the visual focus, the viewing angle of the graphics changes during the presentation.

Re-Presentation of Material Retrieved from the Web

The World-Wide Web opens up a broad range of possibilities for applications using interface agents. One popular vision is the smart and omnipotent *net agent*, which is not only able to read and interpret arbitrary and numerous information sources on behalf of the user, but also filters, condenses and reformulates the retrieved content in order to fuse it into a single comprehensible presentation. This is of high relevance to the user and helps meet her/his preferences concerning the presentation structure, form, and style.

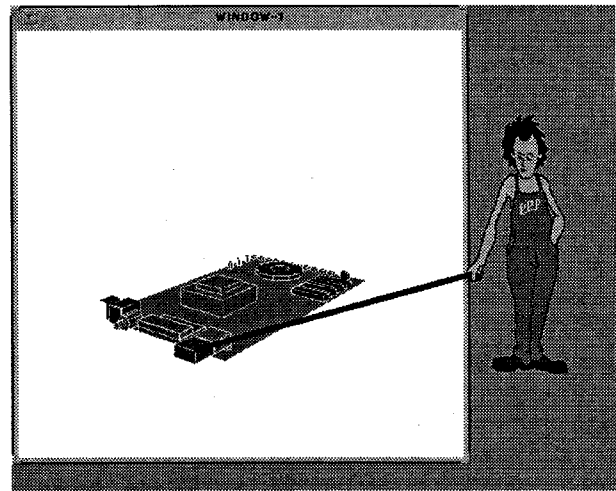


Figure 1: The PPP Persona instantiated as an ...

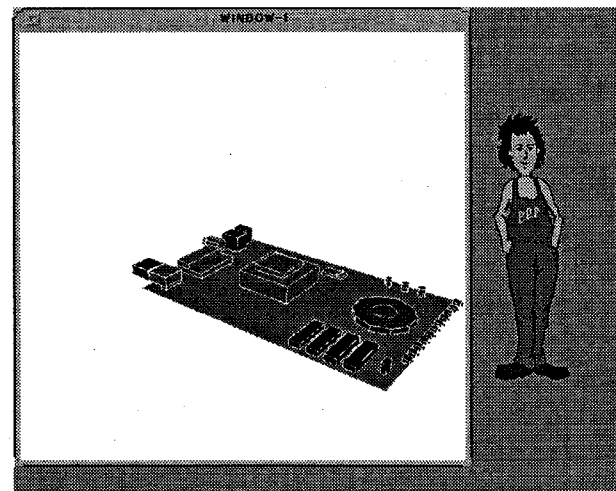


Figure 2: ... animated cartoon character which gives ...

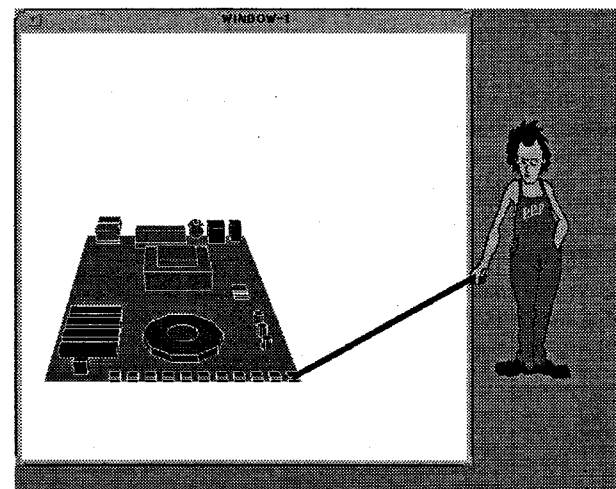


Figure 3: ... an audio-visual presentation

Although this general vision is still far away from reality, there are many application scenarios in which restricted but nevertheless useful interface agents are already technically

feasible.

The next scenario is taken from a study that we conducted for an airline. This time, the PPP Persona acts as a virtual travel agent that presents travelling packages, configured on the fly from a continuously changing stream of online data (available flights, seats, accommodations and so on). The screen shot in Fig. 4 shows the PPP Persona pointing to a picture of an aircraft and suggesting that the user should book a seat in the business class. The screen shot also shows that the appearance of the Persona is not restricted to cartoon characters only. Similar to the option of choosing a favourite text font, a system may provide the user with alternative characters in order to meet his/her individual preferences. This time, the presentation system personifies itself as a “real” person (one of the authors) composed of grabbed video material.

In case graphics is generated automatically, as in the modern example, the presentation system can also build up a reference table that stores the correspondence between picture parts and domain concepts. Since scanned pictures are used in the travelling agent application, such a reference table has been set up manually in order to enable pointing gestures to that material. However, in many cases, the author of a web page already did the job of relating image regions to concepts. For example, many maps available on the web are already mouse-sensitive. That is, the representation formats of these maps comprise specifications of regions which are usually named with the corresponding geographic locations, e.g., names of towns, regions, countries etc. This information can be exploited to enable verbally commented pointing gestures to entities on the map as well. In the travel agent scenario, we use this technique, among other things, to answer a user's request for more information on a flight destination. First, the presentation system checks the availability of a map that includes the flight destination (i.e. the city name) as a mouse-sensitive region. If this is the case, it follows the associated link to fetch the available information. Next, the retrieved data is filtered according to a selected user's interest profile. This information gathering phase is then followed by a presentation phase in which the Persona points to the particular map region and conveys the filtered information verbally (cf. Fig. 5). It is obvious that such presentations cannot be stored in advance, since the author would have to anticipate the current needs of each potential user. For example, one user may be interested in the cultural events in a particular city whereas the other would like to know which cities host computer fares.

It should be mentioned that in this application scenario, the agent is “owned” by the flight company; that is, although the Persona is downloaded to the client side, its behavior and the presentation material will be determined by the presentation system running on the company's web server.

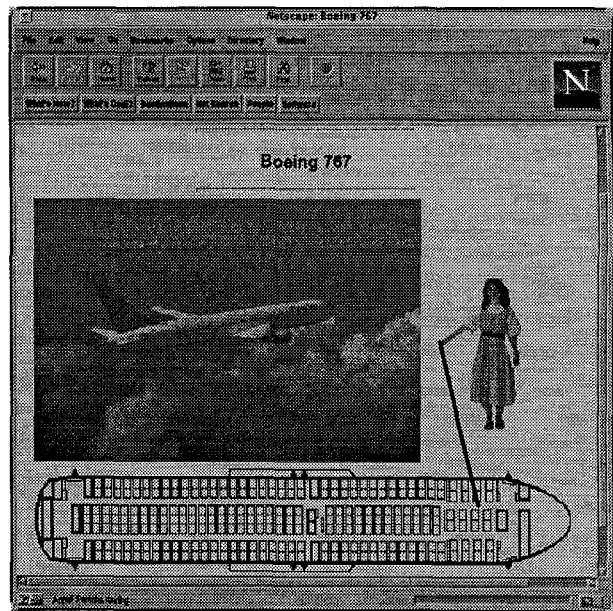


Figure 4: The PPP Persona as a virtual travel agent

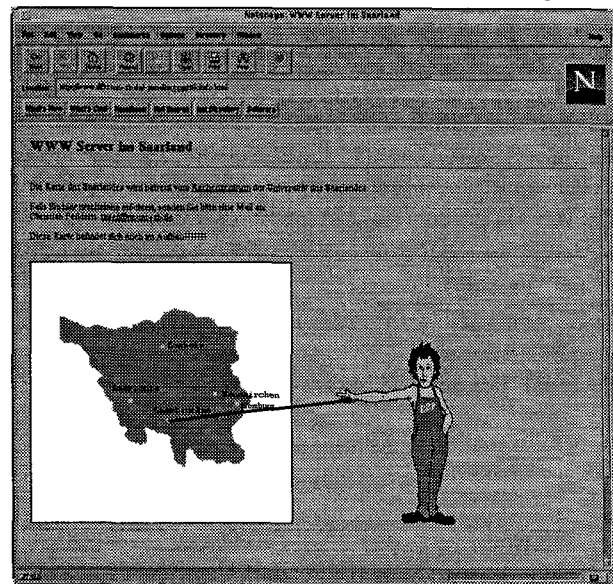


Figure 5: The PPP Persona pointing to map locations

AUTOMATED PLANNING OF PERSONA PRESENTATIONS

Since the PPP Persona doesn't rely on any prestored presentation sequences, we also need a mechanism for composing multimedia material and designing a script for presenting this material to the user.

In our previous work on the presentation system WIP, we developed principles for describing the structure of coherent text-picture combinations (cf. [3]). Essentially, these principles are based on a generalization of speech act theory [25] to the broader context of communication with multiple media, and an extension of RST (Rhetorical Structure Theory, [17]) to capture relations that occur not only between pre-

sensation parts realized within a particular medium but also those between parts conveyed by different media. We also showed how to operationalize a plan-based approach for the automated generation of multimedia presentations on the basis of this framework. However, to cope with the dynamic nature of presentations made by an animated interface agent, several extensions became necessary. These include the distinction between *production* and *presentation acts*, the explicit treatment of the *temporal dimension*, and the definition of *presentation strategies* the interface agent might follow.

Whereas production acts refer to the creation/retrieval of material, presentation acts are display acts, such as *Display-Text*, or acts carried out by the PPP Persona, e.g. *Point*. To run the presentation in an intelligible way, the presentation acts have to be temporally coordinated. For example, pointing to an object depiction requires the exposure of the window containing the depiction. In addition, the pointing gesture should be maintained while a comment on the respective object is being uttered.

To build up a coherent and temporally coordinated presentation for a specified presentation goal, the PPP system exploits design knowledge. In our approach, we use so-called presentation strategies to represent that knowledge. As presentation goals are usually formulated at a high level of abstraction, e.g. *Explain how to operate the modem* or *Provide information about the flight destination*, they have to be refined into appropriate subgoals which can eventually be achieved via production acts and/or presentation acts. The listed strategy (S1) illustrates how such a presentation strategy may look like:

(S1) **Header:** (Describe Persona User ?info-type ?object)
Constraints:
 (And (Bel Persona (I-ISA ?object
 GEOGRAPHICAL-ENTITY))
 (Bel Persona (Contains ?map ?object))
 (Bel Persona (Contains ?page ?map))
 (Bel Persona (URL ?page ?url)))
Inferiors:
 ((A1 (S-Open-Window Persona User ?url ?window))
 (A2 (S-Position Persona User))
 (A3 (Elaborate Persona User ?info-type ?object ?url))
 (A4 (S-Wait Persona User)))
Qualitative:
 ((A1 (before) A2) (A2 (before) A3) (A3 (before) A4))
Metric: ((5 ≤ Duration A4))
Start: A1
Finish: A4

Among other things, this strategy can be used in the scenario of the introduction, in which the user has asked for more information about a flight destination. Assuming that the flight destination is identical with the location saarbruecken-1, the PPP system is confronted with the new presentation goal: (*Describe Persona User Sites saarbruecken-1*). Starting from that goal, the system searches for applicable presentation strategies, i.e., strategies whose Header-entries match

the presentation goal and whose constraints are satisfied. In the example S1 applies since ?object is a geographical entity (first constraint) for which a map exists (second constraint) which is accessible via the web (last two constraints). According to the acts listed in the Inferiors-slot, the system has to access the URL for the map and opens a window (S-Open-Window). After that, the PPP Persona is requested to move to the window (S-Position) and to provide additional information about the object (Elaborate) considering the user's current interests represented by ?info-type. The refinement of *Elaborate* may result in several pointing gestures and speech acts. Furthermore, the strategy includes temporal constraints on and between acts. Qualitative temporal constraints, e.g., (*A1 (before) A2*), are represented in an "Allen-style" fashion (cf. [2]) while quantitative constraints appear as metric (in)equalities. For example, the constraint (*5 ≤ Duration A4*) causes the PPP Persona to wait at least 5 time units before continuing with any other task in order to ensure that the window with the map is noticed by the user.

When designing a presentation, PPP builds up a temporal constraint network which is checked for consistency by computing the numeric ranges on each interval endpoint, the difference between endpoints and all possible Allen relationships between each pair of intervals. Finally, a partial schedule is constructed by resolving all disjoint temporal relationships between intervals and computing a total temporal order. Since the temporal behavior of presentation acts may be unpredictable at design time, the schedule is refined at runtime by adding new metric constraints to the constraint network.

OVERVIEW OF THE PPP SYSTEM

The generation process sketched above has been implemented in the PPP system. Fig. 6 shows the major system components: a *presentation planner*, *medium-specific generators*, currently for graphics [24], text [15] and gestures, the *Persona Server* and a constraint-based *layout manager* [13].

The presentation planner [3] is responsible for determining the contents of multimedia material, selecting an appropriate medium combination and designing a script for presenting the material. This includes an appropriate handling of the timing constraints which are specified in the presentation strategies (for details cf. [4]). Elementary production acts are sent to the corresponding generators which, in turn, inform the presentation planner when they have accomplished their tasks and how they have encoded a certain piece of information.

The results of the generators are taken as input for the design of the presentation script which is forwarded to the display components for execution. The task of the layout manager is the determination of effective screen layouts. The Persona Server (see the next section) carries out Persona actions which, among other things, includes assembling appropriate animation sequences. Both display components signal when

they have accomplished their tasks and inform the presentation planner about the occurrence of interaction events, such as mouse-clicks on windows or the Persona.

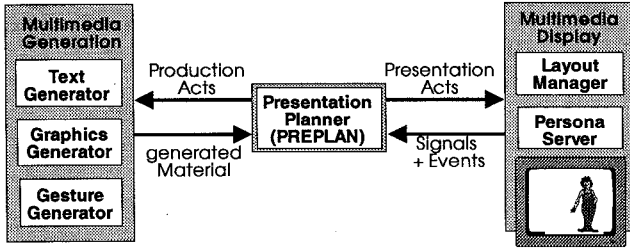


Figure 6: The Architecture of the PPP System

If the PPP system has to provide services for more than one user, we assign to each user/client c_1 to c_n an own PPP-process (PPP_{c_1} to PPP_{c_n}). As Fig. 7 illustrates, the communication between the clients and the PPP-processes is supported by a common gateway interface (CGI), and all processes can share the same information source which may be a database (DB), a more structured knowledge base (KB), or even material fetched from the web. However, for web applications, we use a slightly different system configuration as the one sketched in Fig. 6. The main difference is that each client has to download a presentation display unit, which comes as a Java applet. An instance of this unit includes a Persona server and the display modules of the layout manager.

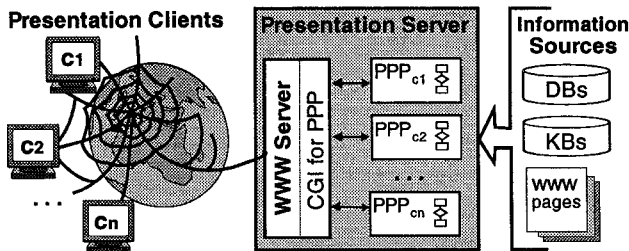


Figure 7: The web version of the PPP System

THE PPP PERSONA SERVER

There are several requirements an animated presentation agent has to meet. According to its functional roles in a presentation (e.g. as in the discussed examples), the Persona must be conversant with a broad variety of presentation gestures and rhetorical body postures. Furthermore, it should adopt a reasonable and lively behavior without being distracting. From the technical point of view, the piece of software that realizes the Persona should be highly independent of application and platform. The next subsection gives a classification of the actions the Persona has to perform. After that, we turn to the technical realization of the PPP Persona server.

Classification of Persona Actions

Since we aim at a generic presentation agent, we are most interested in domain independent actions the Persona should perform. The current repository of such general actions comprises the following types:

- *High-level presentation acts*

This group of actions includes pointing gestures. A pointing gesture is described as a tuple of the form: $\langle \langle \text{pointing-device} \rangle, \langle \text{target-of-pointing} \rangle, \langle \text{pointing-shape} \rangle \rangle$. The Persona's hands/arms and sticks are typical pointing devices. The target refers to the particular entity (text element, graphical object, window etc.) to which the Persona points. Punctual pointing, underscoring, and encircling are possible pointing shapes. Further high-level presentation acts are speaking and the expression of emotions, e.g. getting angry or tired. We assume that all high-level presentation acts are delivered by superordinate components such as a presentation planner in the PPP system.

- *Idle-time acts*

To achieve a lively and "natural" behavior of the Persona, it would not suffice to perform presentation acts only. A distinguishing feature of the PPP Persona is that it even "stays alive" in an idle phase. Typical acts to span pauses are breathing, thumb twiddling and so on. In contrast to high-level presentation acts, idle-time acts are typically not performed on request by an external component. Rather, they are part of a self-monitored, basic behavior of the Persona.

- *Reactive Behaviors*

In case of an interactive system, the Persona should be able to react to some user interactions immediately. For example, if the user moves the window to which the Persona is currently pointing, the consistency of the pointing gesture has to be restored as soon as possible, e.g. by prolonging the pointing stick or by moving the Persona to a new position. Like idle-time acts, this type of reactive behavior belongs to the basic repertoire of Persona's behaviors.

- *Low-level navigation acts*

This class of actions has been introduced for both ontological and economical reasons. From an ontological point of view, it is quite natural to separate a pointing gesture from a preceding navigation act (e.g. moving to a certain position specified relative to a window) which enables the execution of the pointing gesture. The economical motivation for such decompositions lies in the fact that common subactions can be factored out.

- *Basic postures/acts*

While the action types above are semantically motivated, the distinction between basic and non-basic postures/acts is structural in nature. Any action of one

of the types mentioned above is characterized as being basic if it cannot be decomposed into less complex subactions. Technically speaking, a basic posture either corresponds to a single frame of the Persona (e.g. stored as a pixmap), or to an uninterruptable sequence of several frames.

As in many other ontologies of actions, the concept of action decomposition is complemented by the concept of *specialization*. For example, the act “r-hand-lift” is part of the decomposition of the superordinate “r-stick-pointing” act, which in turn is a specialization of the more general act “point-to”. In case an action has several specializations, the current context is taken into account when making a choice. Fig. 8 illustrates the concepts of action specialization and action decomposition.

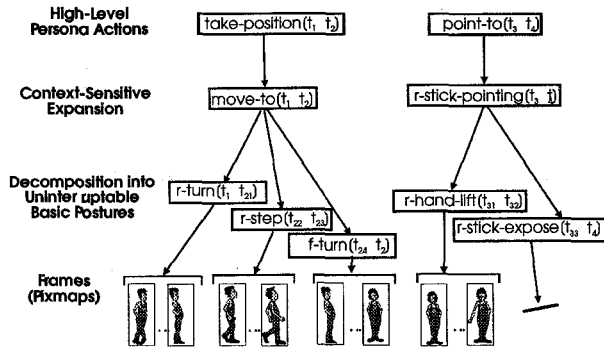


Figure 8: Action types the PPP Persona can perform

We associate with each action a time interval in which the action takes place. For example, the act *take-position* has to be executed during $(t_1 t_2)$. The same applies to the *move-to* act, the specialization of *take-position*. The intervals associated with the subactions of *move-to* are subintervals of $(t_1 t_2)$ and form a sequence. That is the Persona first has to turn to the right during $(t_1 t_{21})$, then take some steps during $(t_{22} t_{23})$ and finally turn to the front during $(t_{24} t_2)$. While the time intervals $(t_1 t_2)$ and $(t_3 t_4)$ are part of the server's input, all subintervals are determined by the server during the specialization and decomposition of the actions.

Architecture of the Persona Server

Let's now turn to the architecture of the PPP Persona server (cf. Fig. 9). Requests for performing high-level presentation actions are received via the application interface. Since it depends on the Persona's current state as to whether or not a request can be immediately handled, they are buffered in an input queue. In return, confirmation is sent back to the application after a task has been performed. Application clients within the PPP system are the PPP presentation planner and the layout manager. The platform interface bridges to the underlying window system, and to several other external devices such as speech generators (for different languages), and an audio player. Vice versa, interaction events recognized by the window system and return values of the external devices are received by the platform interface.

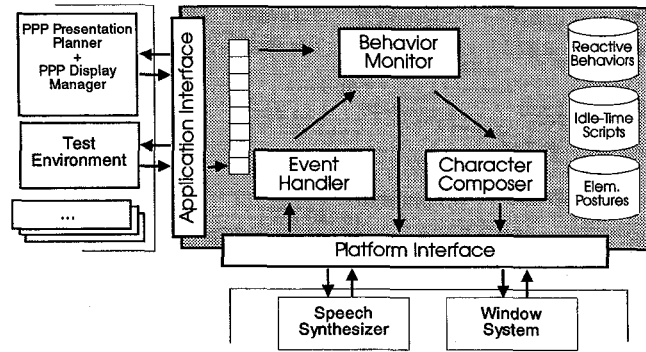


Figure 9: Architecture of the Persona Server

The inner components of the server are a *behavior monitor*, an *event handler*, and a *character composer*. The task of the event handler is to recognize whether input derived from the Persona needs immediate responses from the Persona. That is, the event handler checks for each input message whether the message triggers one of the so-called “reactive behaviors” stored in an internal knowledge-base. If this is the case, the selected behavior is made accessible to the behavior monitor. The task of the behavior monitor is to decide which action to execute next. For instance, if the Persona has no other tasks to perform, it will run an idle-time script that is selected from an internal knowledge-base.

Server Implementations under X11 & Java

Implementations of the PPP Persona server are currently available for Unix platforms running X11, and Java-enhanced WWW-browsers. In the X-version, the Persona server builds upon the X11 library and the X11-Shape extension (cf. [22]). The shape extension allows for the definition of non-rectangular regions in an otherwise invisible window. To use this feature for the graphical realization of the Persona, we first create an invisible window that covers the whole screen. This ensures that in case this window lies on top of the window stack, all other windows below still remain activated for mouse and keyboard input. Second, single postures of the Persona as well as the requisites such as pointing sticks are drawn into the invisible window. Since items drawn into the invisible window remain invisible unless the regions they cover were masked before, we create bitmasks of the same shape as the items (cf. middle part of Fig. 10). For the Persona postures (images or video frames), these masks are computed in a preprocessing phase, and are stored in the X-Server's memory. Bitmasks for other items, e.g. pointing sticks, are computed only on demand at runtime.

For WWW applications, the user first downloads an instance of the Persona server in form of a Java applet. The most expensive part of the loading operation is the transfer of the frames for composing the Persona animations. However, an incremental transfer reduces the loading time that a user perceives. That is, after a sufficient number of frames have been

loaded, the Persona server can be launched and continuously download frames while it is already running. In contrast to the X-windows version, the spatial action ratio of the Persona is restricted to the Java canvas of the web-page. The animation is simply done by bitplotting the corresponding frames onto the canvas. Since Java supports transparency no additional masking is required as in the X version. Once a client-side Persona server has been loaded, the net-load it causes can be neglected, since only highlevel (i.e small textual) specifications of presentation acts have to be provided by the Presentation Server (cf. Fig. 7).

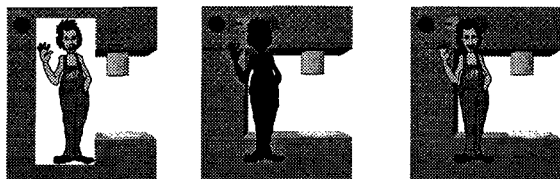


Figure 10: Using a rectangular window (left) vs. the X11 shape extension

RELATED WORK

Animated interface agents and presenters (based either on video material or graphics) have already been proposed by other authors, eg. [16, 20], and can be found in a number of applications (cf. Guides 3.0 [10], Microsoft's "Bob" [18]), and in hypermedia product advertisement (e.g., see the CD-ROM on the new Mercedes Benz S-Class). The purpose of our work was, however, to develop a skilled presentation agent independent of a particular application. This goal is shared with some other approaches:

Gibbs and colleagues [12] focus on technical issues that arise when realizing "Video Actors" by mixing video clips of real persons with the output of an X-server. In their terminology, our realization relies on "server-resident" mixing. However, since we use the widely available X-Shape extension for drawing the single frames corresponding to Persona postures and actions, two advantages are gained. First, we do not need to incorporate dedicated digital video extensions into the server component, as suggested in their paper. More important is the fact that we are not restricted to overlaying the Persona on the X-server output only. Rather, the Persona layer itself is a window which can take on arbitrary positions in the window stack, and can even be moved up and down the stack. Technical similarities also exist between the PPP Persona server and the "Animation Server" described in [7] since both approaches are based on the server/client paradigm, i.e. it is assumed that client application(s) can send requests for animation tasks to the server component. Although developed for different purposes - i.e. the audio-visual illustration of how a user can interact with an application's window-based user interface, in their case, and the audio-visual execution of presentation acts in our case - both approaches are potential augmentations of window-based interfaces. Recent work on 3D-facial animation (e.g.

[14, 23, 19, 21]) also deserves mentioning in this context since this line of research will enable the realization of realistically looking 3D-agents on user interfaces in the near future.

The visual appearance of an interface agent is, however, only one side of the coin. Taking the AI perspective, issues such as the agent's communicative skills and its social behavior are even more important. These issues are explicitly addressed in some research projects, too: The system EDWARD (cf. [8]) was an early attempt to extend a graphical user interface with an animated icon with which the user could communicate via the keyboard in Dutch. A more advanced approach to emulate a face-to-face conversation as it occurs between humans was carried out at Sony CSL (cf. [19]). Their synthetic agent accepts speaker-independent continuous speech input, and also considers visually sensed non-verbal actions such as eye-contact and body movements when interpreting the speech input. Vice versa, the agent interacts with the human user by voice accompanied by facial expressions, head and eye movements. Closely related to the work at Sony is Microsoft's Persona project in which the interface agent is a parrot named Peedy (cf. [5]). Nevertheless Peedy is an anthropomorphic character since it interacts with the user in a natural-language dialogue, and also mimics some non-verbal (human) communicative acts, e.g., Peedy raises a wing to the ear in case speech recognition fails. Since Peedy is to act as a conversational assistant (at least for the sample application, a retrieval system for music CD's), the system comprises of components for processing spoken language, dialogue management and the generation of audio-visual output. "Herman the Bug" is a further 3D-agent (cf. [26]). It has been designed as a pedagogical agent which is part of an interactive tutoring system.

In contrast to these projects, our work concentrated on the operationalization of presentation skills. Unlike all other approaches in which output generation relies on text templates and canned presentation scripts, the PPP system not only generates text and pictures but also automatically builds up temporally coordinated scripts for displaying presentations. Buchanan and Zellweger [9] present a time model for multimedia documents. However, a human author has to specify the material to be presented and the desired temporal relationships between the single document segments from which a consistent schedule is computed. A first attempt of incorporate time into an automated presentation system has been made by Feiner and colleagues (cf. [11]). But, they only investigate how temporal information can be conveyed by dynamic media and don't present a mechanism for synchronizing them. Furthermore, their approach doesn't allow for the specification of metric constraints.

CONCLUSION

We presented two application scenarios in order to support our claim that animated presentation agents are likely to become an integral part of future interfaces of many applica-

tions since they enhance a system's communication abilities. Compared to others who advocated the idea, our emphasis was on the engineering aspect. The usefulness of an interface agent doesn't only depend on the quality of its visual appearance. More important are the underlying mechanisms that determine an agent's behavior. We have sketched an approach for the automated design of presentations that can be performed by an interface agent. The novelty of our system is that it not only designs the multimedia material (such as text paragraphs and graphics), but also plans how to present the material in a temporally coordinated manner. We believe that automated presentation design is a necessary prerequisite since the manual scripting of agent behaviors is simply not flexible enough and also very time-consuming.

An important component of the system is the so-called Persona server which monitors the behavior of an animated presentation agent. It not only enables the execution of complex presentation acts, but also implements a basic behavior independent of the applications it serves. This basic behavior comprises *idle-time actions*, and *immediate reactions* to events occurring in the user interface. Both action types have to be supported in order to obtain a lively and appealing presentation agent. Due to a built-in mechanism for action specialization and decomposition the Persona server accepts presentation tasks at a high-level of abstraction. This feature is particularly useful for web applications since it helps to reduce the net load.

Our current implementations (i.e., the X- and the Java version) provide a good starting point for future work on animated interface agents. Possible directions include: *augmentations* with respect to the Persona's behavior (e.g., the Persona could express emotions as suggested in [6]), the *evaluation* of possible appearances and behaviors of the Persona through different users/user groups, and *further applications* (e.g. Persona can adopt the role of a synthesized shop assistant or investment consultant on the web, or may participate in an audio visual teleconference on behalf of a human user).

ACKNOWLEDGMENTS

This work has been supported by the BMBF under grant ITW 9400. Thanks to Peter Rist for drawing the basic Persona cartoons.

REFERENCES

- 1 B. Adelson. Evocative Agents and Multi-Media Interface Design. In *Proc. of the UIST'92 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 351–356, Monterey, CA, U.S.A., 1992.
- 2 J. F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, 1983.
- 3 E. André and T. Rist. The Design of Illustrated Documents as a Planning Task. In M. Maybury, editor, *Intelligent Multimedia Interfaces*, pages 94–116. AAAI Press, 1993.
- 4 E. André and T. Rist. Coping with temporal constraints in multimedia presentation planning. In *Proc. of AAAI-96*, volume 1, pages 142–147, Portland, Oregon, 1996.
- 5 G. Ball, D. Ling, D. Kurlander, J. Miller, D. Pugh, T. Skelly, A. Stankosky, D. Thiel, M. van Dantzich, and T. Wax. Lifelike computer characters: the persona project at microsoft. In J.M. Bradshaw, editor, *Software Agents*. AAAI/MIT Press, Menlo Park, CA, 1996, to appear.
- 6 J. Bates. The Role of Emotion in Believable Agents. *Communications of the ACM*, 37(7):122–125, 1994.
- 7 K. Bharat and P.N. Sukaviriya. Animating User Interfaces Using Animation Servers. In *Proc. of the UIST'93 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 69–79, Atlanta, GA, U.S.A., 1993.
- 8 E. Bos, C. Huls, and W. Claassen. Edward: Full integration of language and action in a multimodal user interface. *International Journal of Human-Computer Studies*, 40:473–495, 1994.
- 9 M.C. Buchanan and P.T. Zellweger. Automatically Generating Consistent Schedules for Multimedia Documents. *Multimedia Systems*, 1:55–67, 1993.
- 10 A. Don, T. Oren, and B. Laurel. Guides 3.0. In *CHI-93, Video Proceedings*, 1991.
- 11 S. K. Feiner, D. J. Litman, K. R. McKeown, and R. J. Passonneau. Towards Coordinated Temporal Multimedia Presentations. In M. Maybury, editor, *Intelligent Multimedia Interfaces*, pages 139–147. AAAI Press, 1993.
- 12 S. Gibbs, C. Breiteneder, V. de Mey, and M. Papathomas. Video Widgets and Video Actors. In *Proc. of the UIST'93 (ACM SIGGRAPH Symp. on User Interface Software and Technology)*, pages 179–184, Atlanta, GA, U.S.A., 1993.
- 13 W. H. Graf and S. Neurohr. Constraint-based layout in visual program design. In *Proceedings of the 11th International IEEE Symposium on Visual Languages (VL'95), Darmstadt, Sep. 5–9, 1995*.
- 14 P. Kalra, A. Mangili, N. Magnenat-Thalmann, and D. Thalmann. Smile: a Multilayered Facial Animation System. In T.L. Kuni, editor, *Modeling in Computer Graphics*. Springer, New York, NY, 1991.
- 15 A. Kilger. Using UTAGs for Incremental and Parallel Generation. *Computational Intelligence*, 10(4):591–603, 1994.
- 16 B. Laurel. *The Art of Human-Computer Interface Design*. Addison-Wesley, New York, 1990.
- 17 W. C. Mann and S. A. Thompson. Rhetorical Structure Theory: A Theory of Text Organization. Report ISI/RS-87-190, Univ. of Southern California, Marina del Rey, CA, 1987.
- 18 Microsoft. Microsoft Bob. Report, Microsoft, Redmond, WA, 1995.
- 19 K. Nagao and A. Takeuchi. Social interaction: Multimodal conversation with social agents. In *Proc. of AAAI'94*, pages 22–28, Seattle, WA, 1994.
- 20 C.S. Nass, J. Tauber, and R. Ellen. Computers are Social Actors. In *Proc. of CHI-94*, pages 72–77, Boston, MA, 1994.
- 21 H. Noser and D. Thalmann. Synthetic Vision and Audition for Digital Actors. In *Proc. Eurographics'95 14(3), Maastricht*, 1995.
- 22 K. Packard. X11 Nonrectangular Window Shape Extension Version 1.0, X11 R5. Technical report, MIT X Consortium, MIT, Cambridge, Massachusetts, 1989.
- 23 C. Pelachaud. Functional decomposition of facial expressions for an animation system. In M. F. Costabile, T. Catarci, and S. Levialdi, editors, *Advanced Visual Interfaces (Proceedings of AVI '92, Rome, Italy)*, pages 26–49. World Scientific Press, Singapore, 1992.
- 24 T. Rist and E. André. Incorporating Graphics Design and Realization into the Multimodal Presentation System WIP. In M. F. Costabile, T. Catarci, and S. Levialdi, editors, *Advanced Visual Interfaces (Proceedings of AVI '92, Rome, Italy)*, pages 193–207. World Scientific Press, Singapore, 1992.
- 25 J.R. Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge, England, 1980.
- 26 B.A. Stone and J.C. Lester. Dynamically sequencing an animated pedagogical agent. In *Proc. of AAAI-96*, volume 1, pages 424–431, Portland, Oregon, 1996.
- 27 W. Wahlster, E. André, W. Finkler, H.-J. Profitlich, and T. Rist. Plan-Based Integration of Natural Language and Graphics Generation. *AI Journal*, 63:387–427, 1993.