A Lightweight Approach for Experimenting with Tangible Interaction Metaphors

Otmar Hilliges, Christian Sandor, Gudrun Klinker
Technische Universität München
Lehrstuhl für Angewandte Softwaretechnik
hilliges|sandor|klinker@in.tum.de

INTRODUCTION

Interaction techniques for Augmented Reality user interfaces (UIs) differ considerably from well explored 2D UIs, because these include new input and output devices and new interaction metaphors such as tangible interaction.
For experimenting with new devices and metaphors we propose a flexible and lightweight UI framework that supports rapid prototyping of multimodal and collaborative UIs. We use the DWARF [7, 2, 3] framework as foundation. It allows us to build highly dynamic systems enabling the exchange of components at runtime.

OUR APPROACH

Our framework is a UI architecture described by a graph of multiple input and output and control components (User Interface Controller UIC). Furthermore the framework consists of a communication protocol specified in a taxonomy for input tokens [10] and a loose collection of commands [8] that is currently consolidated into a taxonomy.
The input components emit tokens which are received by the UIC. It does a rule based token fusion consisting of *MediaAnalysis* and *CommandSynthesis*. Finally new command tokens are sent to the output components.
We use Petri Nets to model interactions - as is common practice in the area of workflow systems [1].
A Petri Net consists of places, tokens, arcs and transitions. The arcs connect places and transitions. Places and arcs may have capacities. A transition fires when all places at the end of incoming arcs contain (enough) tokens. Transitions execute actions when fired.
Optionally all arcs can have guards on both ends. Guards can define constraints on the type and number of the tokens as well as on the value of the tokens. Transitions only fire when all guards evaluate to true, meaning that all constraints are fulfilled.
In our approach transitions are used to encapsulate atomic interactions. More complex interactions can be modelled by combining several transitions. Our framework allows developers to define rules for *Media Analysis* and *Command Synthesis* and thus modeling interaction in a combination of XML and Java. The rules are encapsulated in the actions performed when transitions fire.

EXAMPLES

In this section we present, in increasing complextity, three examples and their corresponding Petri Nets.

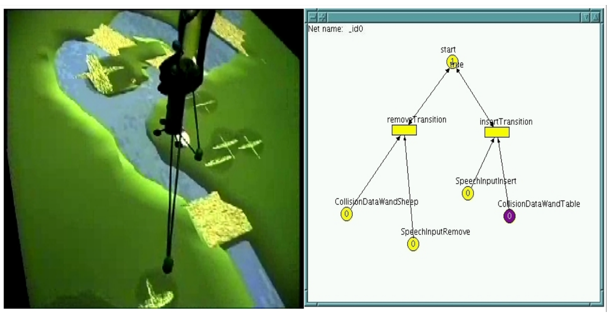

Figure 1: Point-and-Speech interaction in SHEEP. On the left the tangible pointing device, on the right the corresponding Petri Net.

The first example demonstrates how a simple multimodal interaction is modelled in our framework. Figure 1 shows a Petri Net from the SHEEP [9] application, where incoming speech events (”insert Sheep”) and collision events (tangible pointing device hits table) are tokens.
After all places on incoming arcs are filled up, a transition fires. This generates a command object which is sent to the output components. There it leads to the creation of a new sheep.

The second example (see figure 2) demonstrates another simple interaction, but with additional constraints on the flow of events to guarantee consistency. Several sheep can be picked up from the table and dropped back later. Let n be the number of scooped sheep. Then $n > 0$ has to be true, before a sheep can be dropped back onto the table. This *causal constraint* is realized within the guard of the transition which fires the command for dropping a sheep.
Another constraint, realised within the guard for picking up a sheep, is that after a sheep has been dropped, it cannot be picked up again within a certain ammount of time. This *temporal constraint* prevents the user from unwantedly picking

up the dropped sheep with the same interaction. These are very simple examples for constraints. Since the guards logic is implemented in Java more complex constraints are possible.

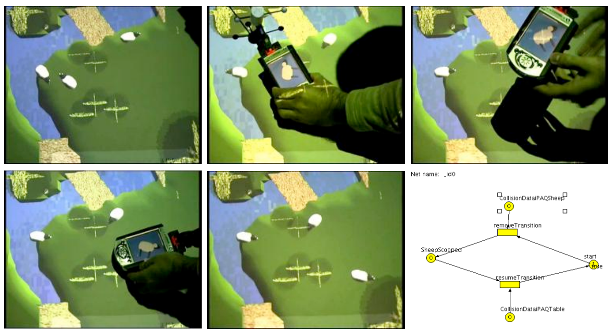

Figure 2: Sequence of images for Scoop-and-Drop interaction with a virtual sheep and an iPAQ. In the lower right corner, the corresponding Petri Net is shown.

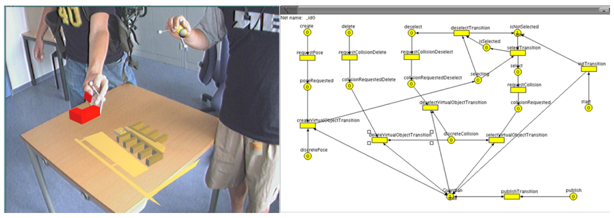

Figure 3: Complex interactions within a tangible modelling application for architects. The corresponding Petri Net is shown on the right.

Within the ARCHIE [5] project, a complex architectural modelling application has been implemented.
The Petri Net for all necessary interactions is shown in figure 3. It enables the user to create, move and change 3D objects (e.g. walls) through multimodal and tangible interactions.
Later those objects can be selected and deselected so that their properties can be changed.
In addition to the complex interactions in this application, the concept of private and public spaces [4] has been explored:
Some of the resulting visualizations can only be seen in the Viewers of the user executing them (*private space*) until he decides to publish his applied changes and thus makes them visible in the Viewers of all users (*public space*).
In the first case the command objects are only sent to the user's Viewer, in the second case they are sent to all Viewers.
Which of these two mechanisms is used is defined within the transitions that belong to these interactions.

DISCUSSION

The framework has been implemented and used in the SHEEP [9] application and several others [6].
It allows an easy composition of multimodal UIs by abstraction of input and output components. Further it allows us to describe interactions based on the formal model of Petri Nets. Additionally temporal and causal constraints on the interactions can be defined as described in the *Examples* section.
The concept of private and public spaces has been addressed too, utilizing the advantages of the DWARF publisher/subscriber middleware architecture.
Finally the simulator delivers an easy to understand visualization during development and runtime. The simulator visualizes the structure of the Petri Nets themselves and how tokens are passed through and transitions fire during runtime. Screenshots can be seen in the figures 1, 2 and 3
However we encountered some limitations. Petri Nets are by nature rigid. That leads to problems when unexpected things happen and dynamic reactions are necessary. Two mayor problems are caused by this.
Error handling is very limited. And all interactions are predefined, that prevents us from building intelligent and learning systems.
We focus on the flexible adaption of components and the easy, intuitive possibility to model and exchange interaction metaphors. We did encounter that this approach is sufficient to model a broad range of multimodal and collaborative interactions.

REFERENCES

1. W. AALST, *The Application of Petri Nets to Workflow Management*, The Journal of Circuits, Systems and Computers, 8 (1998), pp. 21–66.
2. M. BAUER, B. BRUEGGE, G. KLINKER, A. MACWILLIAMS, T. REICHER, S. RISS, C. SANDOR, AND M. WAGNER, *Design of a Component-Based Augmented Reality Framework*, in Proc. IEEE International Symposium on Augmented Reality (ISAR'01), New York, pp. 45–53.
3. M. BAUER, B. BRUEGGE, G. KLINKER, A. MACWILLIAMS, T. REICHER, C. SANDOR, AND M. WAGNER, *An Architecture Concept for Ubiquitous Computing Aware Wearable Computers*, in Proceedings of the 2nd International Workshop on Smart Appliances and Wearable Computing (IWSAWC 2002), Vienna, Austria.
4. A. BUTZ, C. BESHERS, AND S. FEINER, *Of Vampire Mirrors and Privacy Lamps: Privacy Management in Multi-User Augmented Environments*, in ACM Symposium on User Interface Software and Technology, pp. 171–172.
5. DWARF, *The ARCHIE Homepage*. `http://www1.in.tum.de/projects/lehrstuhl/twiki/bin/view/DWARF/ProjectArchie`.
6. DWARF, *Complete list of DWARF based Projects*. `http://www1.in.tum.de/projects/lehrstuhl/twiki/bin/view/DWARF/ProjectsOverview`.
7. DWARF, *The DWARF Homepage*. `http://www.augmentedreality.de`.
8. E. GAMMA, R. HELM, R. JOHNSON, AND J. VLISSIDES, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, 1995.
9. A. MACWILLIAMS, C. SANDOR, M. WAGNER, M. BAUER, G. KLINKER, AND B. BRÜGGE, *Herding Sheep: Live System Development for Distributed Augmented Reality*, in Proceedings of ISMAR 2003.
10. J. WOEHLER, *Driver Development for TouchGlove Input Device for DWARF based Applications*. Systementwicklungsprojekt, Technische Universität München, 2003.