Building User Interfaces with Tasks, Dialogs and XML

Karin Coninx, Tim Clerckx, Bert Creemers, Kris Luyten, Jan Van den Bergh, Chris Vandervelpen
Limburgs Universitair Centrum
Expertise Centre for Digital Media
Universitaire Campus, B-3590 Diepenbeek, Belgium
`http://www.edm.luc.ac.be`
{karin.coninx,tim.clerckx,bert.creemers,kris.luyten,jan.vandenbergh,chris.vandervelpen}@luc.ac.be

ABSTRACT

We present two ongoing research efforts, both aim to support the use of models for designing (multi- and multiple-device) User Interfaces. The first tool, a part of the Dygimes framework, shows how context and tasks can be combined. It allows to generate prototype interfaces from context-sensitive task models. It builds uoon a runtime environment in Java and an XML-based High Level User Interface Description (HLUID) language to generate the prototype interface. The second tool, uiml.net, experiments with another HLUID and another runtime enviroment to generate interfaces. Both tools are work in progress.

SUPPORT FOR CONTEXT-SENSITIVE TASK SPECIFICATIONS IN DYGIMES

Introduction

This tool adds support for transforming regular ConcurTaskTrees task specifications into Context-Sensitive ConcurTaskTrees using the *decision node* approach. First the regular task specification is loaded. The leaves of this specification contain HLUID components that are necessary to perform the actual (sub)task. Then, leaf nodes can be transformed into decision nodes that will be provided with decision XML (information about what context parameters apply on the current node) and subtrees (ConcurTaskTrees task specifications describing the possible tasks for each type of context in the decision XML). Afterwards the corresponding dialog model for the current context will be calculated and the user interface can be deployed.

Case Study

The example shows that transitions can occur between different devices, and that tasks can be spread onto different devices. This will be done by capturing the context in a preprocessing step before deploying the actual user interface. The environment simulates the context information by providing the necessary information in an XML document. Figure 1 shows the tool and a prototype where parts of the user interface are rendered on desktop PC and other parts on a cell phone.

AN UIML RENDERER FOR .NET

Introduction

This part of the demo shows a Uiml renderer for the Mono .Net framework. We focus on the Gtk# widget set, and show how a UI can be built with the renderer. This demo also shows how application logic and the HLUID can be connected together with the rules one can describe in an Uiml document. In contrast with the Dygimes renderer this HLUID offers more functionlity in the language itself, while Dygimes combines different models to generate the User Interface.

Case Study

A simple calculator will be shown, where the UI written in Uiml will be *very loosely coupled* with the application logic. The rendered UI is shown in figure 2.

Different implementations of the application logic can be linked to the UI by the rendering engine. To show this we provide both an implementation that can handle floating point operations, and one that is limited to integer operations. Both implementations are compiled into dynamic libraries and can be passed to the uiml.net renderer as an argument (`calc.dll` for the floating point operations and `calc-int.dll` for integers only). E.g. rendering the calculator example with the floating point support can be done like this: `uiml.net -uiml calculator.uiml -libs calc.dll`.

Second, to show the renderer is easy extensible and does not require changing the code to support a different API, we show how the vocabulary file can be changed and how this is reflected in the rendered UI. For this we use an example with different widgets, like shown in figure 3. For example, we show how to deal with a property being redefined in the API without changing the rendering engine.

REFERENCES

1. Tim Clerckx, Kris Luyten, and Karin Coninx. Generating Context-Sensitive Multiple Device Interfaces from Design. In *CADUI'2004 long paper track*, 2004.

2. Kris Luyten and Karin Coninx. Uiml.net: an Open Uiml Renderer for the .Net Framework. In *CADUI'2004 long paper track*, 2004.

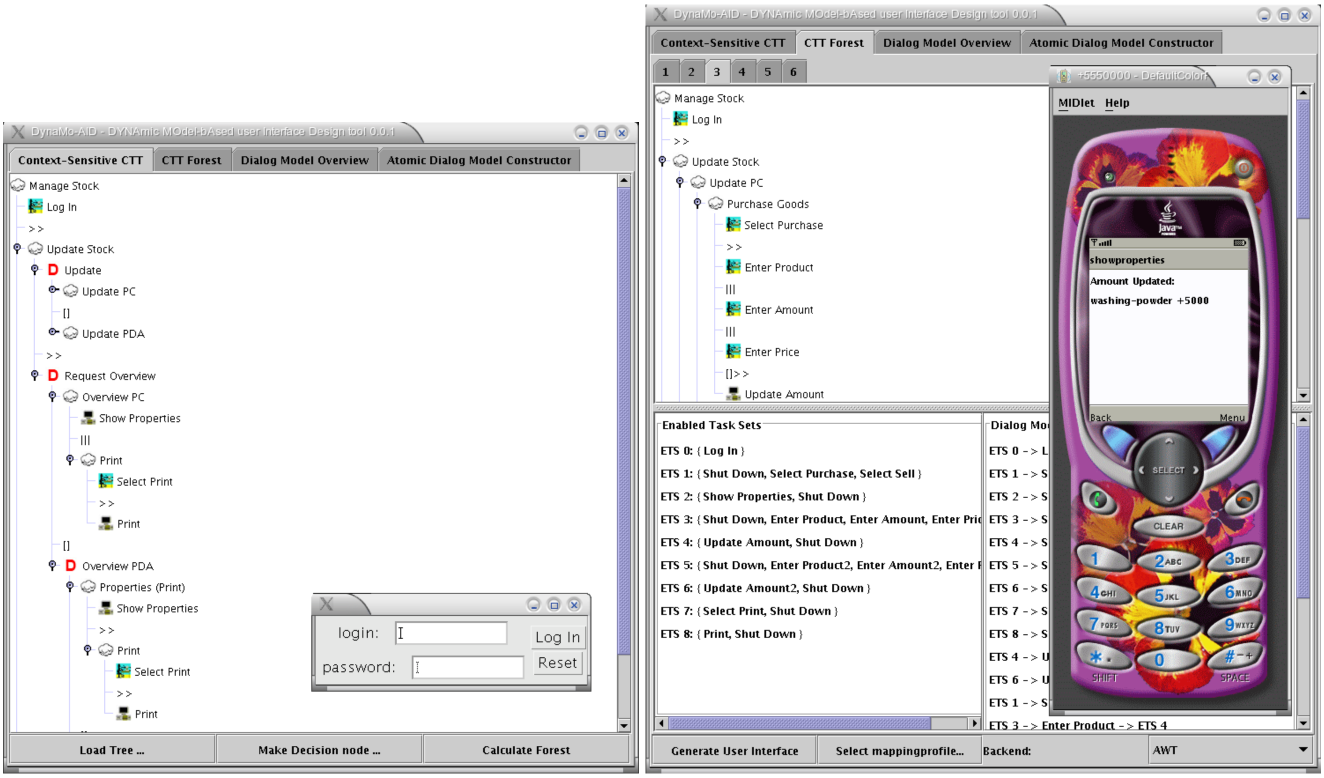

Figure 1. Examples for Desktop PC and Cell Phone

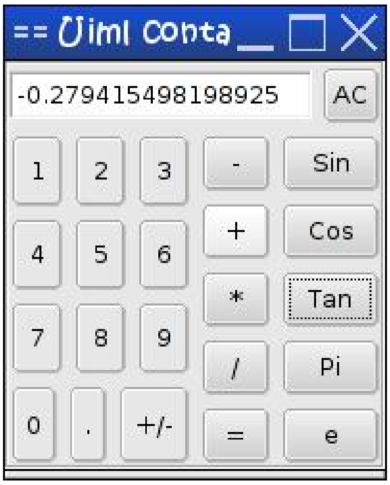

Figure 2. A calculator with Uiml.net

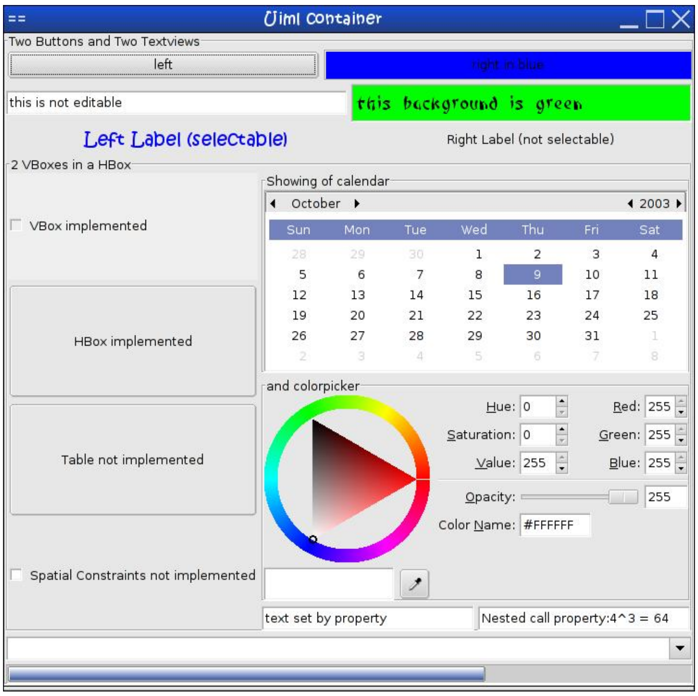

Figure 3: Different widgets rendered with the Gtk# Vocabulary