Specifying User Interfaces for Runtime Modal Independent Migration

Kris Luyten Tom Van Laerhoven
Karin Coninx Frank Van Reeth

Expertisecentrum Digitale Media
Limburgs Universitair Centrum

CADUI, May 17, 2002, Valenciennes

Presentation Overview

1. Introduction
2. Runtime specification generation
3. User Interface migration
4. Adapting to a new environment
5. Requirements
6. The case study
7. Original system
8. Target system
9. Difficulties

10. Future work

11. Questions

1. Introduction

- How to migrate a UI at runtime
- Without any knowledge of the target at design-time
- And no human interference
- Allowing all kinds of devices
- Dynamically extensible

2. Runtime specification generation

2.1. Situation

- Application or service is running
- Somebody wants access
- In another environment
  - Probably using another device

2.2. Proposed Solution

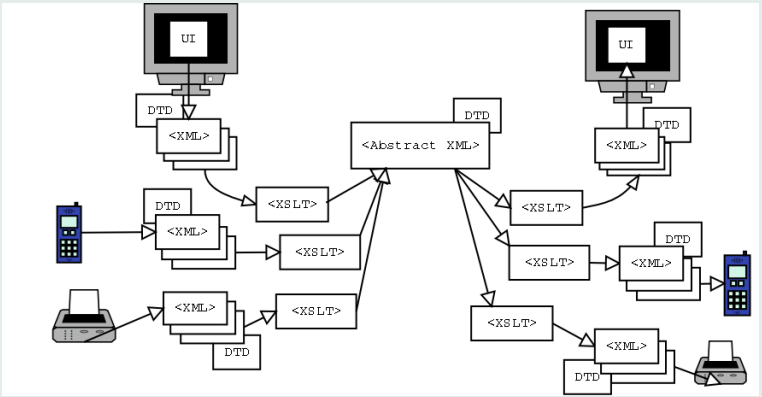
- Request for UI description
- Service generates an UI description *at runtime*
- Using introspection or built-in framework
- The UI description uses the eXtensible Markup Language syntax.

3. Why an XML-based syntax?

- Platform independent
- Declarative
- Consistency
- Constraints
- Extensible
- Reusable
- Transformations

4. User Interface migration

- Transform the XML-based UI description
- Obtain an abstract description
- Concrete Interaction Objects → Abstract Interaction Objects
- Transport description
- Abstract Interaction Objects → Concrete Interaction Objects
- Render the description


UI
DTD
<XML>
<XSLT>
<Abstract XML>
UI
DTD
<XML>
DTD
<XML>
<XSLT>
<XSLT>
DTD
<XML>
<XSLT>
<XSLT>
DTD
<XML>
<XML>
<XSLT>
DTD
<XML>
DTD

- How to handle the application logic?
  - Transfer the application logic
  - Invoke the application logic remotely
- Applications are built using components/component system
  - Component offers service and description of interface/functionality
  - Invoking application logic is location transparent
  - "Invokers" are specified in the resulting UI description
  - Application logic separated from presentation

5. Adapting to a new environment

- Modality
- Widget set
- Device constraints

6. Requirements

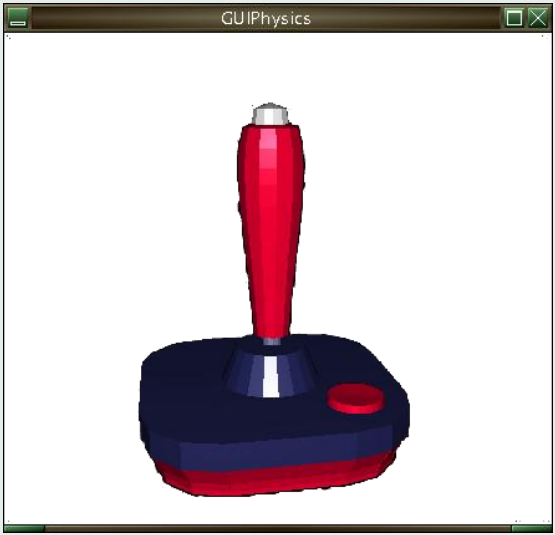
- The UI should be "serialisable" at runtime
  - Using XML syntax
  - According to a predefined DTD or XML Schema
- A set of XSLT documents for transforming the description
- An intermediate format (according to DTD or XML Schema)
- The target should provide a rendering engine
- Some information about the task is necessary

7. A case study

- Two totally different frameworks:
  - Framework for physical simulations
  - Framework for rendering XML-based UIs in 2D
- Can they interchange a User Interface at runtime, without them knowing of each other?
- The service: A camera surveillance system:
  - control the orientation of a camera (up/down, left/right).
  - Take a snapshot
  - "Apply" the orientation

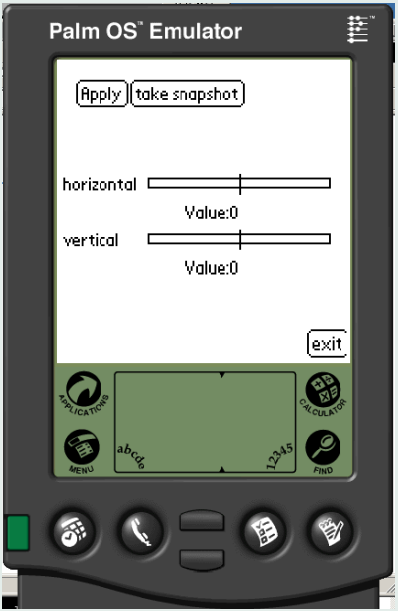
8. The original system

- Framework for physical simulations
- Suitable for building 3D interfaces
- A 3D joystick for controlling the camera
- Uses XGL to describe scenes
- Programmed in C++, with XML serialisation framework


GUIPhysics

9. The target system

- PDA Palm IIIc
- XML-based UI renderer
- Programmed in Java (kAWT)
- several limitations:
  - Screen space
  - Widget set
  - Means of interaction


Palm OS™ Emulator
Apply
take snapshot
horizontal
Value:0
vertical
Value:0
exit
APPLICATIONS
CALCULATOR
MENU
FIND
abcdé
12345

10. Difficulties

- Providing a *Context* description
  - What to include? Task description, user profile,...
  - So many possible situations...
- Making more intelligent decisions about what to transform

11. Future Work

- More complex UIs
- A more general conversion process
- Access with speech interfaces
- Apply this approach to separate components

12. Questions, remarks,. . .

?