

# **PUIR: Parallel User Interface Rendering**

**Kris Van Hees & Jan Engelen**

Katholieke Universiteit Leuven

Department of Electrical Engineering

ESAT - SCD - DocArch



# Agenda

- Introduction
- Related work
- Parallel User Interface Rendering
- Implementation details
- Testing
- Conclusions
- The road ahead...

# Introduction

- GUIs are all around us: computers, MP3 players, home appliances, mobile phones, ...
- Implied visual interaction model poses a complication for users with disabilities (especially blindness)
- Use of multiple graphical toolkits within a single environment complicates matters even more (Linux: Qt, GTK, Athena, ...)
- Existing solutions tend to either face limitations or involve user interface changes (compile-time or runtime)

# Related work

- EU project: GUIB – Screen reader
- Mercator – Screen reader
- HAWK – Non-visual interaction toolkit (in AVANTI)
- SUPPLE – UI adaptation mechanism
- GNOME Accessibility Architecture
- UsiXML – Abstract user interface definitions
- GiTK – Rendering abstract UI definitions

# Related Work:

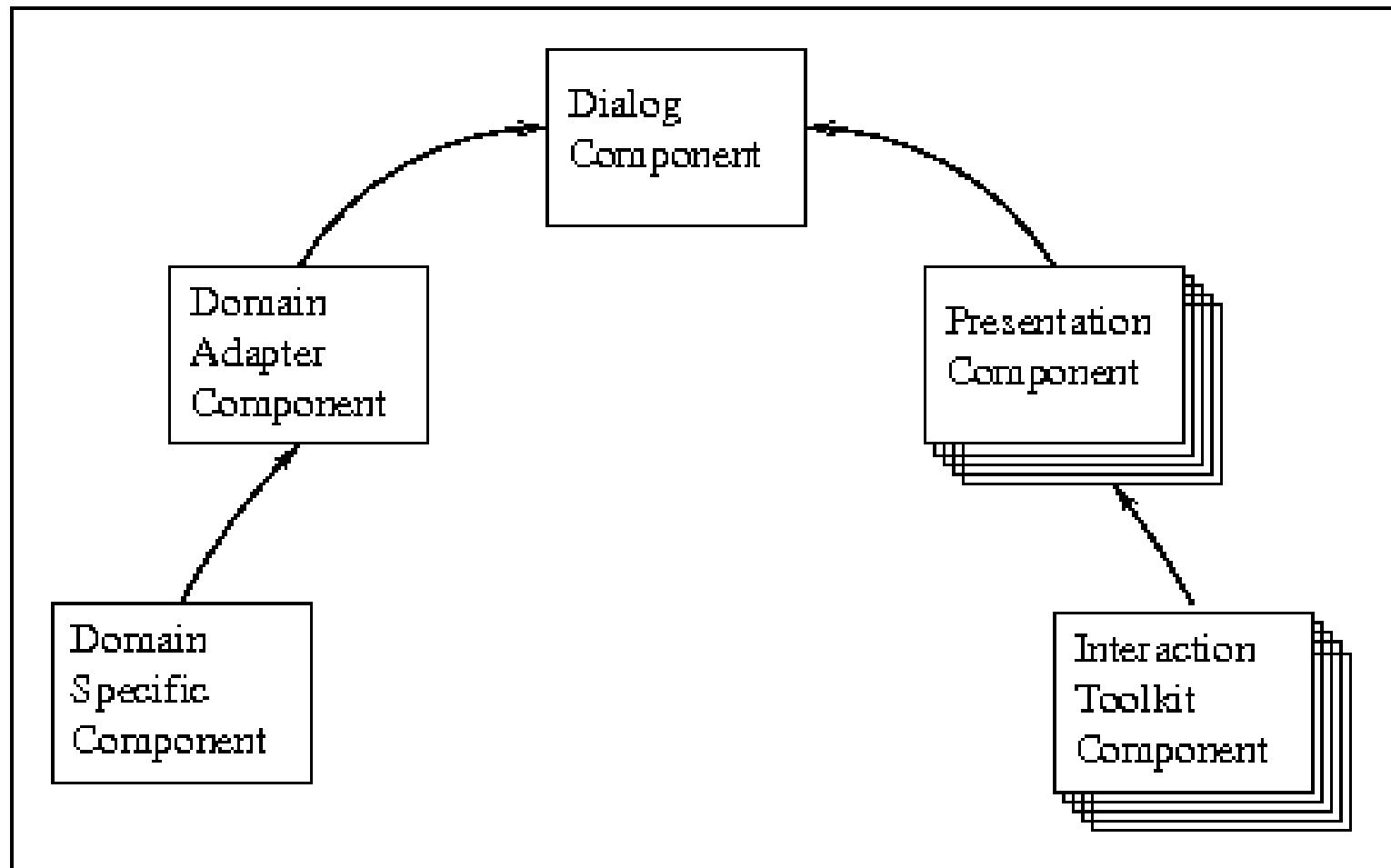
## HCI Issues

- Coherence between visual and non-visual interfaces
- Exploration in a non-visual interface
- Conveying graphical information in a non-visual interface
- Interaction in a non-visual interface
- Ease of learning
- Maintainability

# Parallel User Interface Rendering

- User interface defined in abstract form (XML file)
  - Effectively defines data and how to present data
- Abstract UI description interpreted by an AUI engine
  - Handles user interaction semantics
- UI presentation rendered by modality-specific agent(s)
- Rendering agent depends on system specific toolkits
- Multiple rendering agents can work in parallel
  - Provide both visual and non-visual rendering

# Parallel User Interface Rendering: ARCH model



# Parallel User Interface Rendering: ARCH model (cont...)

- **Domain Specific Component:** application
- **Domain Adaptor Component:** AUI description (XML)
  - Defines the UI
  - May contain rendering specific annotations
- **Dialog Component:** AUI engine
  - Interaction semantics of UI elements
  - Encapsulation of UI data elements
  - UI focus management
  - Event synchronisation

# Parallel User Interface Rendering: ARCH model (cont...)

- **Presentation component:** Rendering agent
  - PUIR uses multiple simultaneous rendering agents
  - Each rendering agent interacts with the AUI engine
- **Interaction Toolkit Component:** Modality toolkit
  - Functionality specific to output modalities
  - Toolkit often also implements specifics of input device handling

# Implementation details:

## AUI Engine

- Application can make calls to elements at the AUI level
- AUI engine dispatches events to application
- AUI engine dispatches events to rendering agents
- Rendering agents can make calls to elements at the AUI level
- AUI engine implements user interaction semantics
- AUI engine implements focus management
- As of now, 25 UI elements have been implemented

# Implementation details:

## Rendering agents

- Implement the API for AUI engine interaction
- Both in-process and remote agents are supported
- All agents receive the same events (in parallel)
- Two proof-of-concept rendering agents:
  - In-process: Swing
  - Remote: Speech synthesizer output (in progress)
    - Uses a stub/proxy approach for transparency and caching

# Implementation details:

## User input

- Three input modalities supports in proof-of-concept:
  - Keyboard input: handled directly by the AUI engine
  - Mouse input:
    - Interpreted by rendering agent in graphical context
    - Translated into semantic events for AUI engine
    - AUI engine acts on the semantic events
  - Braille keyboard input:
    - Interpreted in context and presented as events

# Testing

- Unit testing proven invaluable to ensure consistency
- Manual comparison of functionality between PUIR-based implementation and Swing-based implementation
- Unit testing to verify correctness of proxy UI tree
- Input from users solicited throughout the process
- Still a need for more actual user testing of the proof-of-concept implementation (release forthcoming)

# Conclusions

- Parallel User Interface Rendering
  - Powerful technique for Design-for-All
  - Based on research on abstract user interfaces
  - Parallel presentations rather than a derivative
  - Coherence between renderings
  - Generic: not just for non-visual renderings
  - Supports remote access
  - Can be used in automated application testing as well

# The road ahead...

- Completion of proof-of-concept
- Extensive testing
- Public release

*At all stages, feedback from real users will be crucial!*

**Kris Van Hees & Jan Engelen**  
Katholieke Universiteit Leuven  
Department of Electrical Engineering  
ESAT – SCD – DocArch  
Kasteelpark Arenberg 10  
B-3001 Heverlee  
Belgium

**[kris@alchar.org](mailto:kris@alchar.org), [jan@docarch.be](mailto:jan@docarch.be)**