

# Abstracting the Graphical User Interface for Non-Visual Access

**Kris Van Hees & Jan Engelen**

Katholieke Universiteit Leuven

Department of Electrical Engineering

ESAT - SCD - DocArch



KATHOLIEKE UNIVERSITEIT  
**LEUVEN**



# Overview

- Introduction
- Related work
- HCI issues for non-visual presentation of GUIs
- One way: Gnome Accessibility Architecture
- Another way: Abstract User Interfaces
- Proposed solution
- More advanced problems
- The long road ahead...

# Introduction

- Graphical User Interfaces:
  - More intuitive to sighted users
  - New challenges imposed by visual concepts
- In the past:
  - Mostly focussing on single-platform: MS Windows
  - Remote access for everything else
- Recent years:
  - More platforms (Unix-Linux flavours, Mac OS-X, ...)
  - Multiple graphical toolkits rather than a single uniform one



# Related work

- EU project: GUIB – Screen reader
- Mercator – Screen reader
- Gnome Accessibility Architecture
- Fruit – Abstract widget toolkit
- **UsiXML** – Abstract use interface definitions

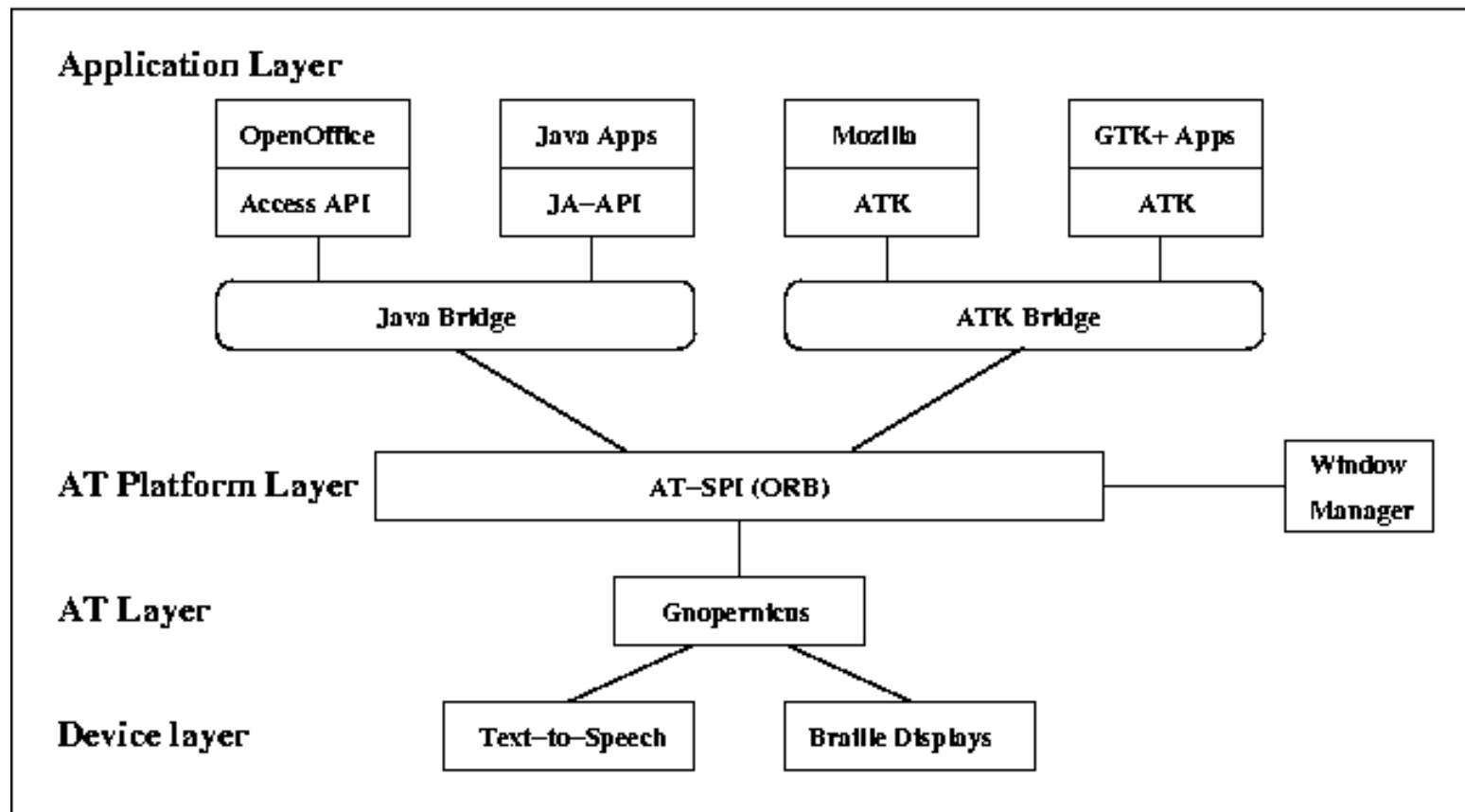
# HCI issues for the non-visual presentation of GUIs

- Coherence between visual and non-visual interfaces
  - Mental interaction model must be substantially similar between visual and non-visual presentations
  - Blind and sighted users should be able to observe each other's interactions
- Exploration in a non-visual interface
  - Access to the content of applications windows is not enough
  - Spatial parameters can carry information

# HCI issues for non-visual presentation for GUIs

- Conveying graphical information in a non-visual interface
  - Graphical object attributes (appearance, style, ...)
- Interaction in a non-visual interface
  - GUIs commonly employ visual idioms (clicking, dragging, sliding, ...)
- Ease of learning
  - ... because otherwise no one would use it!

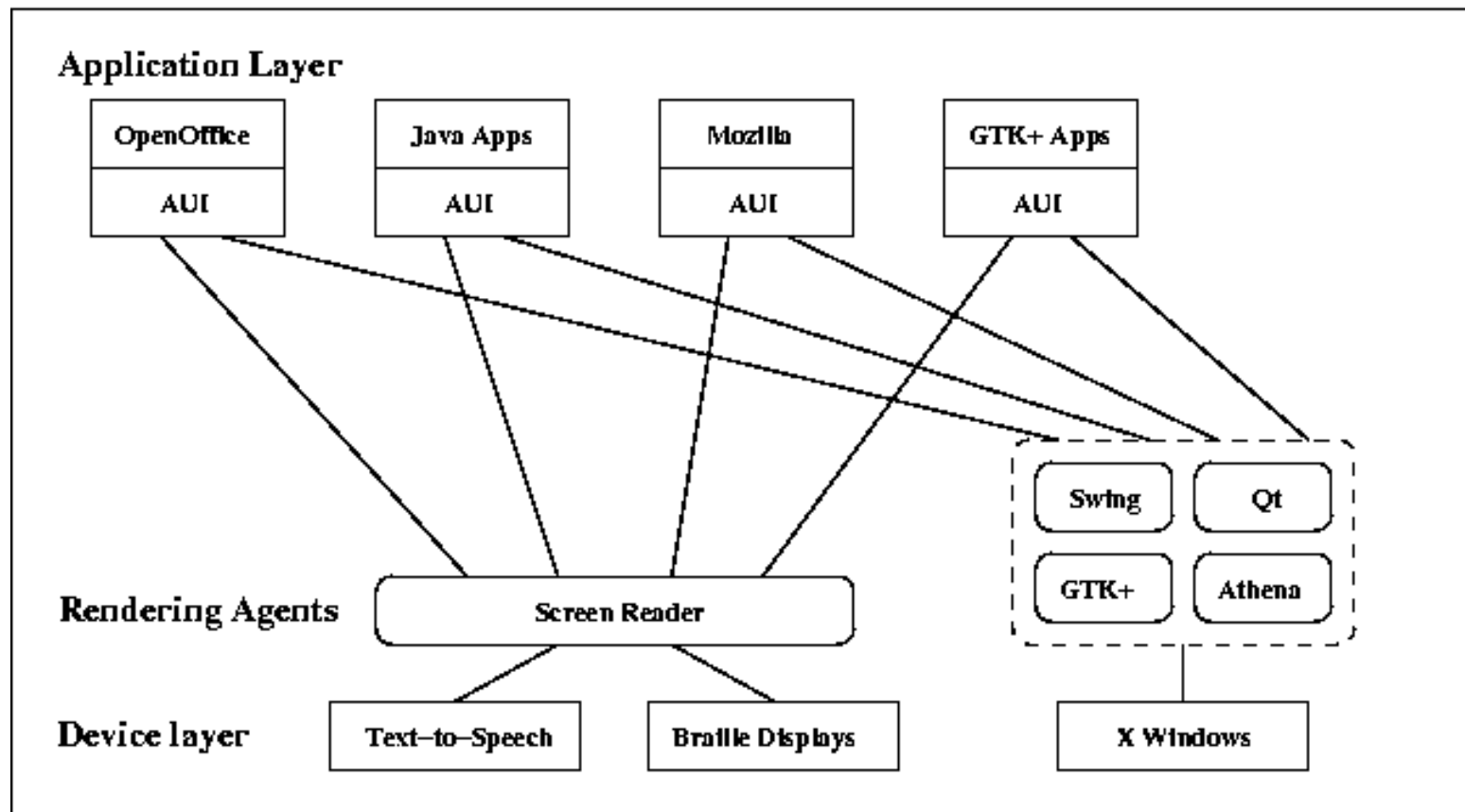
# One way: Gnome Accessibility Architecture



# One way: Gnome Accessibility Architecture

- Applications opt-in to accessibility
- Toolkit-specific APIs interface with AT-SPI by means of bridges
- Accessibility agents interface with AT-SPI to query state information

# Another way: Abstract User Interfaces



# Another way: Abstract User Interfaces

- Applications provide an AUI description
- The UI description is in a standardized format (**UsiXML** or a similar format)
- The UI description can be transformed for a specific output modality
- The UI description is rendered by agents:
  - Graphical toolkit agents for visual presentation
  - Non-visual agents for auditory and/or tactile presentation

# Proposed solution

- Abstract user interface descriptions in UsiXML (or similar form)
- Transformation rule sets
  - Transform the AUI into an alternate AUI
  - Mostly augment the AUI with non-visual attributes and elements
- Rendering agents
  - Transform the AUI into a CUI (Concrete User Interface)
- Device Layer
  - Transforms the CUI into an FUI (Final User Interface)

# More advanced problems

- Dynamic user interfaces
  - Very common (grayed items, last accessed files, ...)
  - AUI definition therefore needs to support runtime updates
- Legacy applications
  - Cannot be avoided
  - Automated reverse engineering can get you somewhere
- So-called “Creative Programming” (usually in games) does not respect rules
  - 100% success rate is virtually impossible

# The long road ahead...

- Implementation of a basic graphical AUI rendering agent
- Implementation of a basic non-visual AUI rendering agent
- Implementation of a transformation rule engine
- Definition of transformation rule sets

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

# Contact information

**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)**