

# Abstract UIs as a long-term solution for non-visual access to GUIs

**Kris Van Hees & Jan Engelen**

Katholieke Universiteit Leuven

Department of Electrical Engineering

ESAT – SCD - DocArch



KATHOLIEKE UNIVERSITEIT  
**LEUVEN**



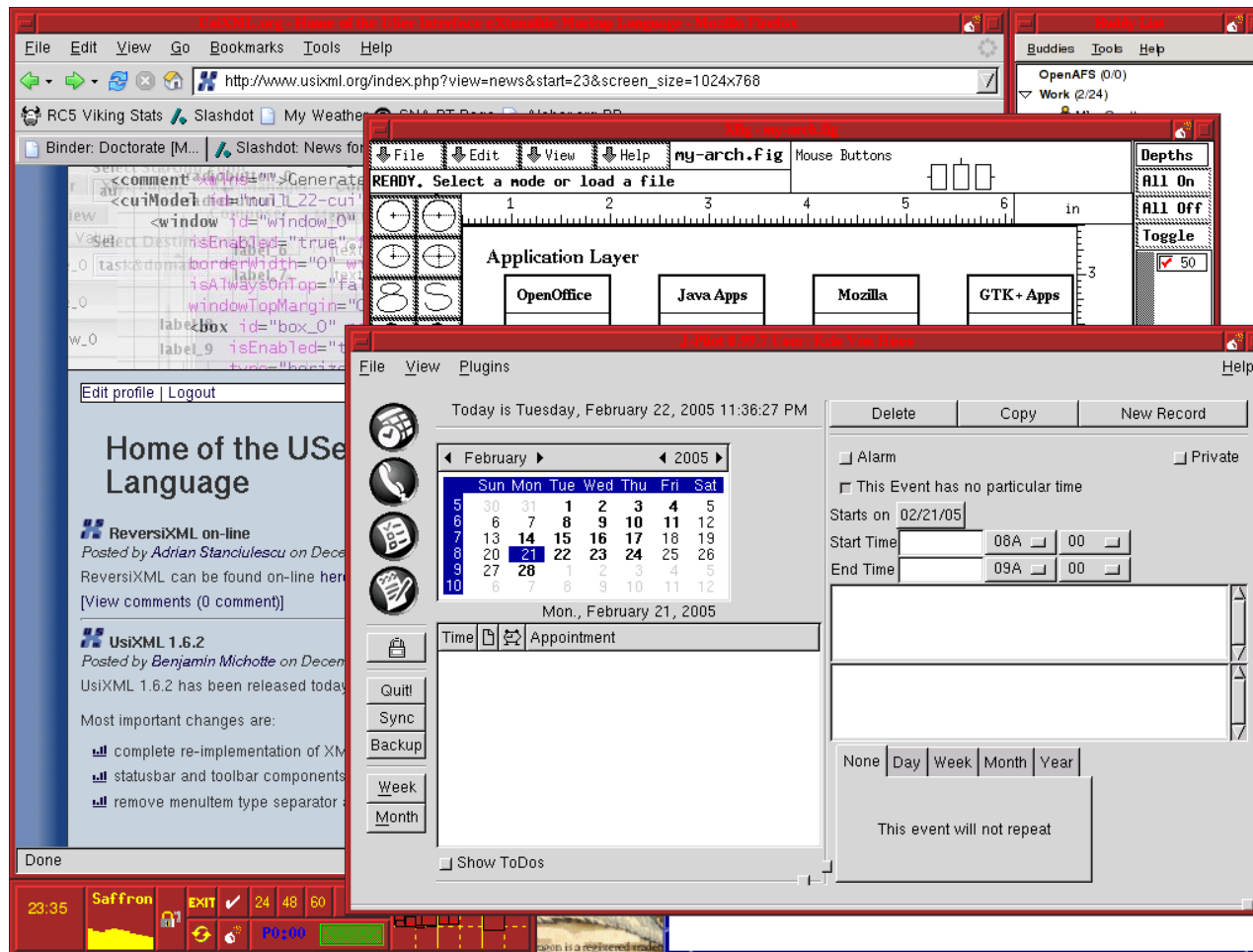
# Agenda

- Introduction
- HCI issues for non-visual presentation of GUIs
- One way: Gnome Accessibility Architecture
- Another way: Abstract User Interfaces
- How do you get from here to there?
- More advanced problems
- The long road ahead...
- Questions?

# Introduction

- “In the beginning”: text-based screen reading
- Traditional Windows desktop environments:
  - Largely consistent user interface toolkit
  - Hooks for screen readers to access system level facilities
- Unix desktop environments:
  - Multiple user interface toolkits
  - Mix'n'match environments are not uncommon

# Sample Unix desktop environment



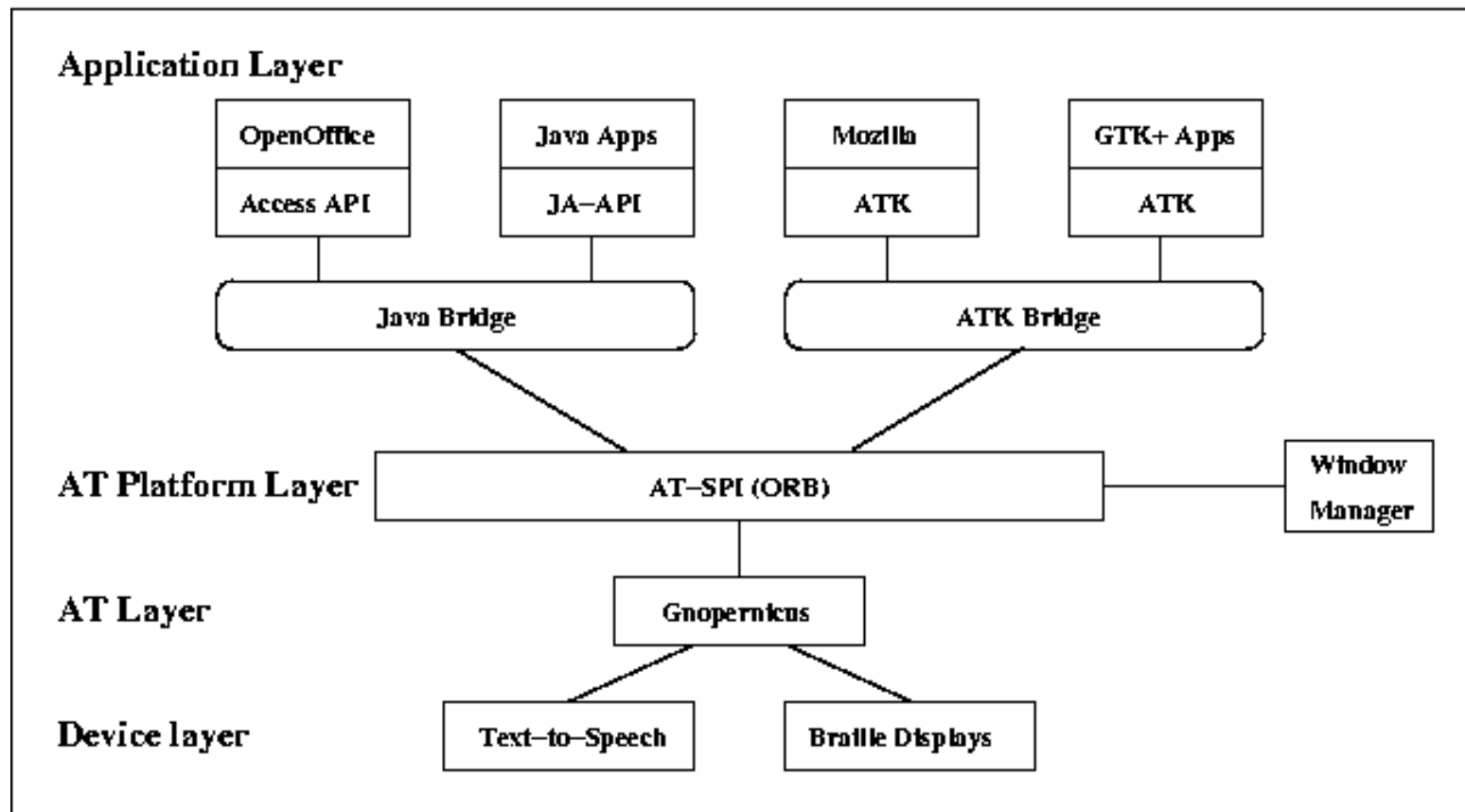
# HCI issues for non-visual presentation for GUIs (1/2)

- 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

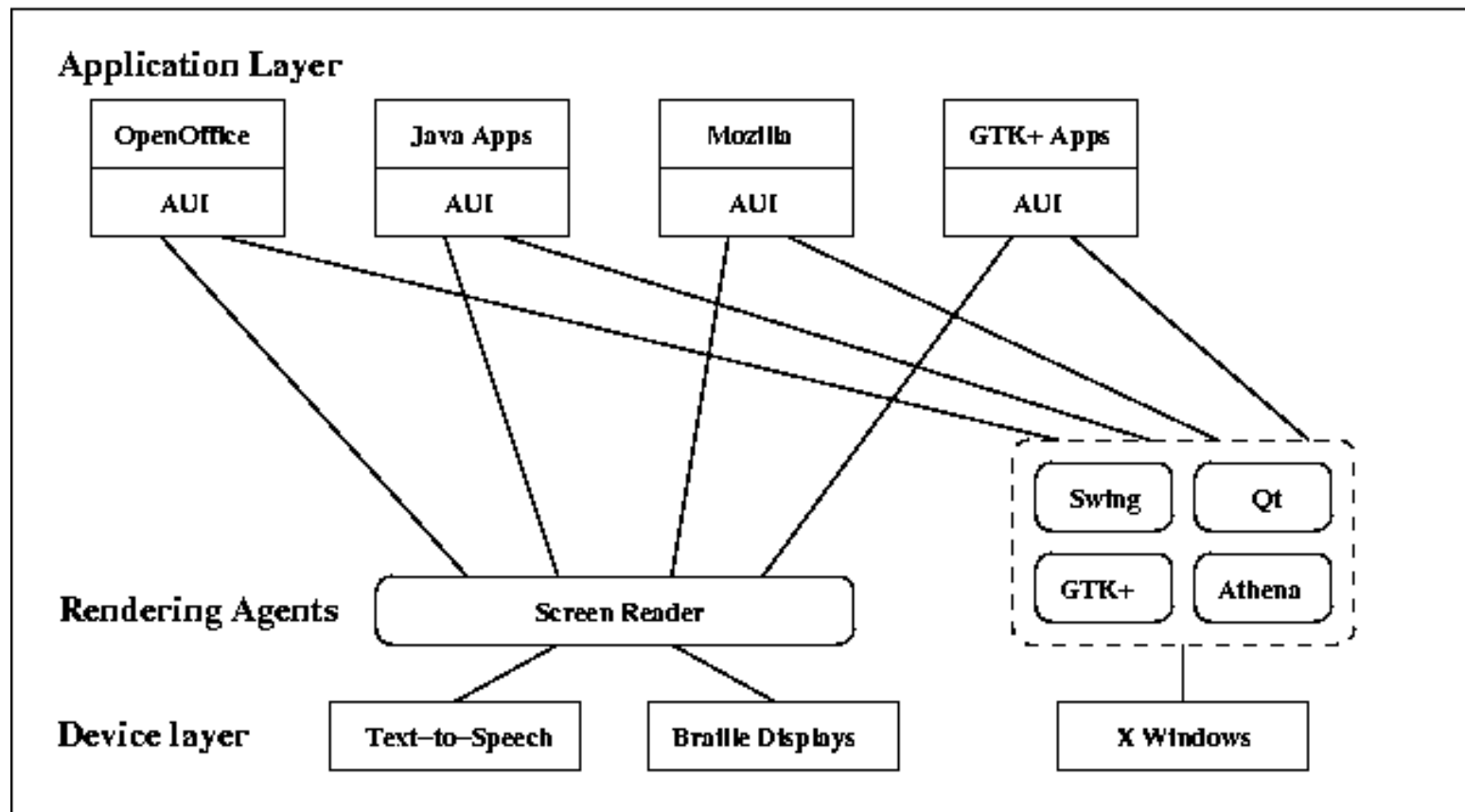
# HCI issues for non-visual presentation for GUIs (2/2)

- 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



# Another way: Abstract User Interfaces



# How to get from here to there?

- Abstract user interface descriptions in UsiXML (or alike form)
- Transformation rule sets
  - Transforms the AUI into an alternate AUI
  - Mostly augmenting the AUI with non-visual elements
- Rendering agents
  - Transforms 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 menu items, last accessed files, ...)
  - AUI definition needs to support runtime updates
- Legacy applications
  - Cannot be avoided
  - Automated reverse engineering gets you somewhere
- So-called “Creative Programming”
  - 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!*

# Questions?

If you have any questions.  
this is the time to ask them!

# 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)