GITK in comparison with other adaptive interface toolkits

Dipl. Inf. Stefan Kost (FH)

HTWK Leipzig, Postfach 30 11 66, D−04251 Leipzig, Germany

kost@imn.htwk−leipzig.de

Abstract: Although a few adaptive toolkits already exist, with GITK (’Generalised Interface ToolKit’) another solution appears. This young solution targets a different range of applications. Research has shown that the complexity the other toolkits supply is not always needed but hinders cross media−domain adaption. GITK has only a few requirements, is light−weight and thus can generate interfaces of a great variety.

Keywords: ’generalised interfaces’, ’adaptive toolkits’, ’multi−dimensional adaption’, ’multi modal interfaces’

1 Motivation

Many of currently available adaptive interface toolkits are specialised to provide user interfaces for single media domain. The reason for it is, that the toolkit wants to use as much features as possible for information representation available in the chosen domain. Complex forms of presentation are unfortunately difficult to replace with equivalent ones of other media domains − sometimes they even do not exist. Other approaches have heavy technical requirements and are therefore not well suited to mobile devices.

Avoiding the above mentioned problems, requires an architecture differing in many details. In my work I develop a technology called 'Generalised Interface ToolKit' (GITK). In contrast to the former approaches, this solution strives to provide a toolkit for applications, which can live with lack of interface objects for complex data presentation. The benefit they get in turn is, that GITK can make those applications very adaptable. Possible adaptions are ranging from text based interfaces with speech output and braille line support to aid the blind, over common graphical interfaces, up to interfaces using a telephone line with speech i/o and touchtones. The architecture of the GITK system would even allow dynamic construction of system interfaces (Interfaces to remote control an application (e.g. via a REXX), by another application).

2 Introduction

2.1 Adaptive interface toolkits

Adaptive interface toolkits equip an application with an interface at run−time. They are able to change certain aspects of the interface (adaption), as the user or the current environment requires it. In my work I subdivide adaption into personal (personal preferences, education, sensorical and motorical capabilities), cultural (language, locale) and technological (i/o hardware) adaption and refer to it as multi−dimensional adaption.

Many existing solutions such as 'User Interface Markup Language' (UIML) (UIML, 2002) or 'eXtensible Userinterface Language' (XUL) (Dakin, 2003) are designed to generate visually oriented interfaces. With this limitation they can avoid many difficulties, such as transforming between several media (e.g. Text to Speech). My research has shown that a large number of application do not really need those features to work. A large quantity of applications, which are in daily use for communication or organisation are mainly using textual information. Text can easily be represented in various media−domains such as graphics or audio and can be entered with a great variety of hardware.

This leads to the idea of restricting the complexity of available specialised presentations in favour of the better transformability.

2.2 The GITK approach

Like other approaches GITK uses an 'eXtensible Markup Language' (XML) for interface description called 'Generalised Interface Markup Language' (GIML). One of the most important considerations made when designing GIML, was the strict separation between functional and presentational description of an interface. The application logic remains in the application itself. Functional description is provided as XML (GIML) files from the application, but presentational settings are derived from 'eXtensible Stylesheet Language' (XSL) files, which come from user and system profiles. These information are merged with the functional descriptions by using 'eXtensible Stylesheet Language Transformations'(XSLT) to form a final interface description. The profile data could come directly from a file−system or from a remote profile server.

A second corner stone is the modular architecture of GITK. The solution consists of several layers and a plugin interface for independent interface rendering modules. The plugins allow easy extensibility. New rendering modules can be developed separately and added at anytime.

The open architecture makes it an ideal testbed for upcoming developments in the field of 'Human Computer Interaction' (HCI). Applications ported to use GITK can immediately tried with new technology or new presentation styles, without that they need to be changed.

3 Current status

The software is still under heavy development, although several demonstrations do already exists. Renderers for text−based interfaces as well as one for a graphical interface are written and ready to be demonstrated.

The GIML has been designed, after exploring the needs of the targeted applications, as well as capabilities and requirements of interfaces.

Currently one or two students will start to write rendering modules to see if the term 'gerneralised' holds.

Everyone can participate in the development on http://gitk.sf.net.

References

UIML (2002), UIML − User Interface Markup Language, *http://www.uiml.org/intro/index.html*, Virginia Tech Corporate Research Center.

Neil Deakin. (14.Apr.2003), XUL Tutorial, *http://www.xulplanet.com/tutorials/xultu/intro.html*, http://www.xulplanet.com.