The Next Generation Graphical User Interface

Intelligent Systems Winter Semester 2013/14

Leroy Ruegemer Rahul Indoria Shoaib Khan

Supervisors: Herwin van Welbergen & Philipp Kulms

Bielefeld University, Faculty of Technology

Abstract

There is advanced Human/Computer interaction methods such as speech or face recognition, virtual agents and other devices are available but experiments and research often concentrates on one technologie. As the evaluation of these is hard, we propose an easy way (even for non-specialists) to create user interfaces that is able easily include the different technologies available to experiment and prototype with next generation interfaces.

Introduction

Technology is evolutionary, it keeps changing. The same goes with user interfaces. The earliest type of interface was called “Command line interface”, where a keyboard was used to access a program in a computer using a set of commands. With the addition of pointing devices (such as the computer mouse) the interfaces evolved to “Graphical User Interfaces”. The input with devices that are no longer restricted to mouse and keyboard but also includes other pointing devices such as a stylus, joystick, and touch screen. The computer device can be mobile, tablet, PDA, MP3 player and even TV.

Now researchers are experimenting the next generation UI. They have several heterogeneous input and output devices available. Integration of these devices in a single interface is difficult especialy for non specialists or researchers of different institutes as the device developers. This results in research that is concentrated on only one of these new techologies.

Main Objectives

The main objective of this project is to create an engine that makes it possible to easily integrate different technolgies into one interface/program. As interfaces are event driven, it is possible to describe the behavior with finite state machines, so that it should be possible to design the interaction using State Chart Extensible Markup Language (SCXML). As the potency of the different technologies like virtual agents rises the engine should be easily expandable.

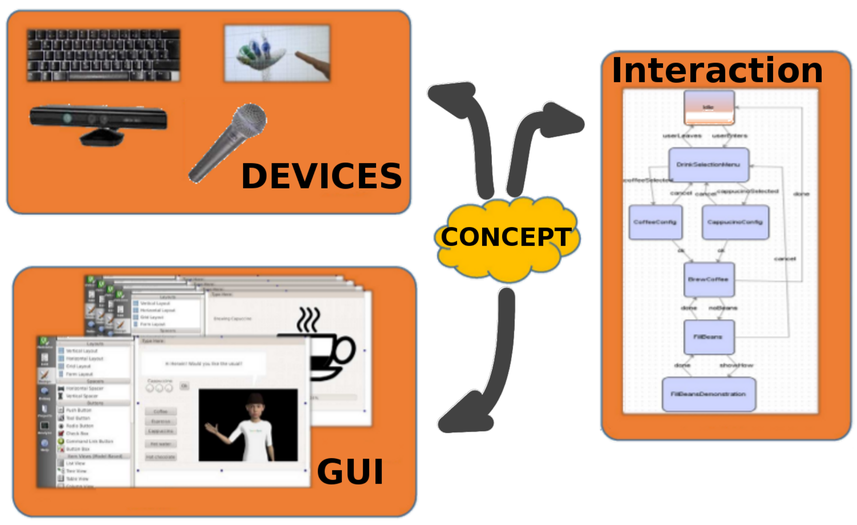

Figure 1: Three parts of next generation user interfaces

The main objectives are:

1. Creating a user friendly way to prototype Graphical User Interfaces.
2. Using SCXML to describe the behaviour of the program.
3. Easy method to add new elements and functionality.
4. Central program which handles communication between elements of the user interface.
5. Maintaining SCXML defined behaviour.

Materials, Methods, Architecture

The materials required to develop the single main program that handles all these heterogeneous devices are:

1. Apache commons SCXML which is used for implementation of an engine aimed at execution of a program behaviour defined by a finite state machine as SCXML file.
2. SCXMLgui, a graphical user interface for editing SCXML state machines.
3. The Ipaaca middle-ware that is used to realize the communication between the SCXML engine and external programs. Ipaaca implementations for Java, Python and C++ are available. Ipaaca uses some incremental unit containing one or more strings to sent messages from an output buffer in one program to the input buffer of another program.
4. The Articulated Social Agents Platform (AsapRealizer) which is the Behaviour Markup Language (BML) realizer for incremental, fluent and multi-modal interaction with a virtual human or robot. BML is used for defining gestures, facial expression and speech performed by a virtual agent.
5. Qt4 is used as a cross-platform interface realizer. The Qt designer is used to generate UI files. These can be loaded to display graphical user interfaces.

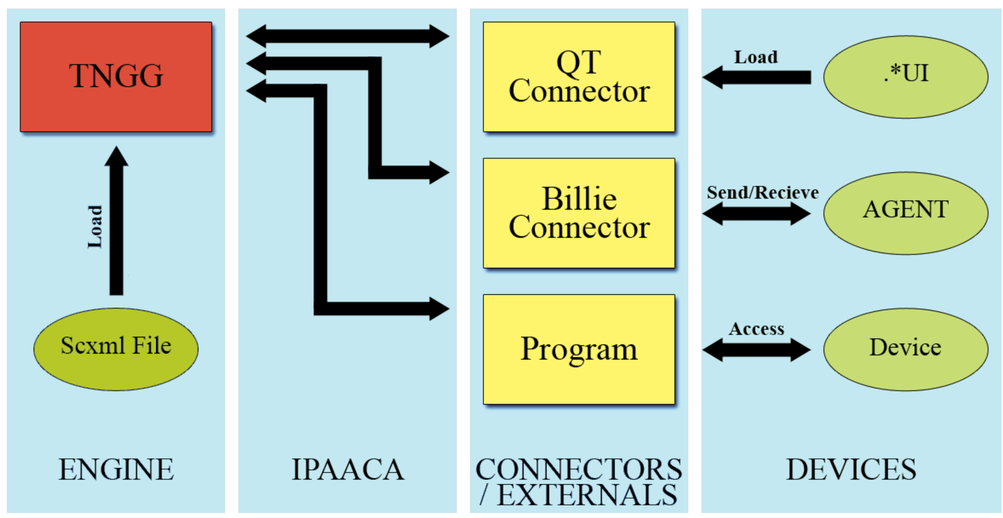

Figure 2: TNGG Architecture

The architecture of the TNGG engine is shown in **Figure 2**. The engine loads a SCXML file which describes the program behaviour and communicates with an external connectors over Ipaaca. Externals are used to hook-up different devices but currently only the agent and Qt are implemented.

Results

To evaluate the functionality of the TNGG engine an interactive coffee machine was used as a use case and designed with SCXMLgui (**Figure 3**). The current state of program is able to run this coffee machine, show UI files created with QTDesigner and display what type of coffe the user ordered.

```
<state id="CoffeeConfig">
  <onentry>
    <ui:open file="coffeeconfig.ui"/>
    <agent:do action="say" arg="How much sugar and milk do you want?"/>
  </onentry>
    <transition event="UI.BTN.OK" target="$BrewCoffee"/>
</state>
```

Listing: CoffeeConfig state

The CoffeeConfig state is shown in **Listing**. If the state machine reaches this state two external functions are executed. The QTGui changes the current GUI by loading coffeeconfig.ui and the virtual agent utters the defined sentence ”How much sugar and milk do you want”.

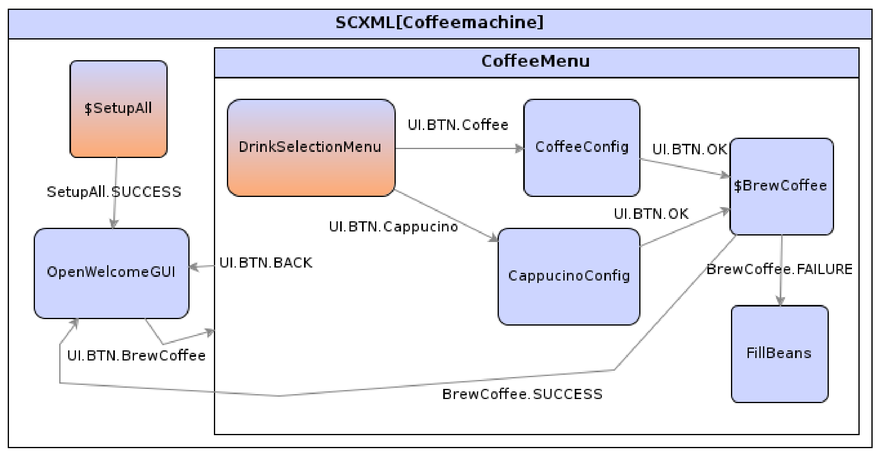

Figure 3: Coffee machine SCXML file

Conclusions

TNGG engine was successfully implemented with all required functionality of user interfaces with heterogeneous devices. The main objectives listed above were successfully implemented. To make a prediction about the productivity and usability for non expert users it is required to design and perform user case studies. Additional external connector should be implemented to increase the basic functionality for users without expertise and documentation must be tailored for non-specialist users.

References

[1] QT Project Developer. Qt4. `http://qt-project.org/`, January 2014.

[2] Apache Software Foundation. Apache commons scxml. `http://commons.apache.org/proper/commons-scxml/`, January 2014.

[3] Human Media Interaction group Twente Sociable Agents group Bielefeld. Asap-project. `http://asap-project.ewi.utwente.nl/`, January 2014.

Acknowledgements

We would like to thank Herwin van Welbergen and Philipp Kulms for their guidance and motivation. Sincere thanks to the SOA group Bielefeld university.