

noBounds – Mobile Device Interoperability

Jörg Brakensiek
Nokia Research Center
Nokia Inc.
Palo Alto, CA, USA
Jorg.Brakensiek@nokia.com

Abstract—Applications and services running on a mobile device can offer extended user-experience if they would be able to seamlessly interoperate with external environments, by exchanging data (e.g. audio, video, display) and taking back command and control. The proposed demonstrator will show different technical alternatives how such interoperability can look like and how they compare with respect to performance and bandwidth requirements.

Keywords—component; Interoperability, Mobile Device, Car

I. NOBOUNDS USE CASE

A mobile device is not designed to operate in an optimal manner in all environments. There are many use cases, where the mobile device would benefit from resources and functionality provided from an external environment. One prominent example for this is the car domain. Assuming personal navigation software, running locally on the mobile device, it would benefit from interoperating with the car as shown in Fig. 1. The end-user relevant content (i.e. turn-by-turn directions, the map data and additional information) is created and rendered on the local device and streamed to multiple displays. At the same time the application is controlled via available links in the car.

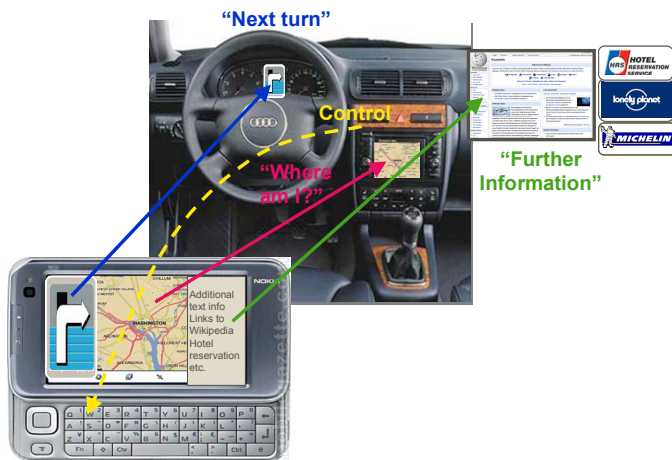


Figure 1. Mobile Device Interoperability within a Car

Other uses cases in this domain are streaming multimedia content (audio and video) to in-built screens and the audio system, and productivity applications. In latter case this will be enabled via the connection of high-resolution displays to the mobile device.

II. NOBOUNDS ARCHITECTURE

The noBounds demonstrations will show different architectural options relevant for the mentioned use cases. In general, basic connectivity is provided through USB, BT or WLAN. We are using IP as a generic (i.e. bearer independent) transport mechanisms, as shown in Fig. 2. In our demonstration setup we enabled connectivity via TCP/IP over wired USB [1].

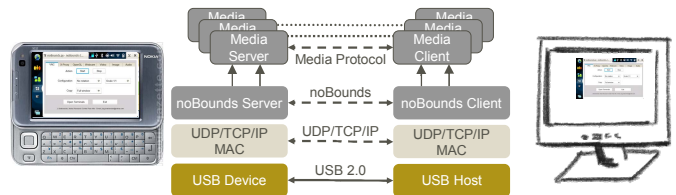


Figure 2. noBounds Basic Architecture

The noBounds demonstrator setup consists of a basic Client-Server architecture, which will synchronize, start and control different media servers. These media server are implementing the interception.

There are different options, where the interceptions between the local device and the remote device can happen, as shown in Fig. 3. On the one extreme, the local frame buffer is copied to the remote side, leading to a very thin remote implementation; on the other extreme, encoded video/audio data is streamed, which require decoding capabilities on the remote device. Composition of the window scene (e.g. OpenGL, X11) on the local device and final rendering on the remote device is somewhat sitting in the middle. The chosen interception layer is mainly defining required bandwidth on the link as well.

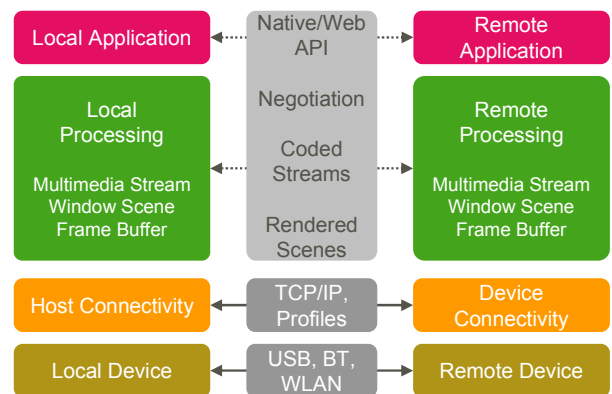


Figure 3. noBounds Interception Options

III. NOBOUNDS DEMONSTRATIONS

The proposed demonstration will show three main technology options in this setup. The demonstration setup is based on a Nokia N810 Linux based Internet Tablet device [2] and a Linux based Laptop as the remote rendering device. This concept is in early research and develop phase and we would like to use the demonstration to enable discussion and trigger feedback with the audience on the relevancy and acceptance of the different technologies, as this interoperability challenge certainly require a broader industry consensus.

A. Remote Frame Buffer

We will demonstrate the copy of mobile device's frame buffer into the remote device, as shown in Fig. 4. As nearly all UI processing is done locally, this setup will show very low performance needs on the remote side, at higher bandwidth requirements on the USB link. This concept is completely transparent to the applications and the windowing system.

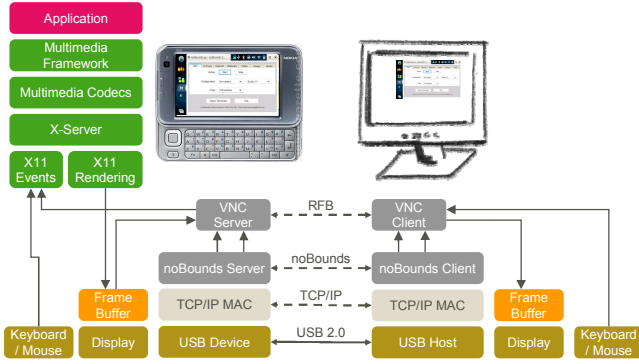


Figure 4. noBounds – Remote Frame Buffer Demonstration

B. X-Server

We will demonstrate the use of the X-Protocol as a suitable interception point, which enables remote displaying transparent to the applications (Fig. 5). We will demonstrate different options, using standard X-protocol, X-extension (like GLX for OpenGL rendered scenes), and an SSH setup with X11 forwarding [4].

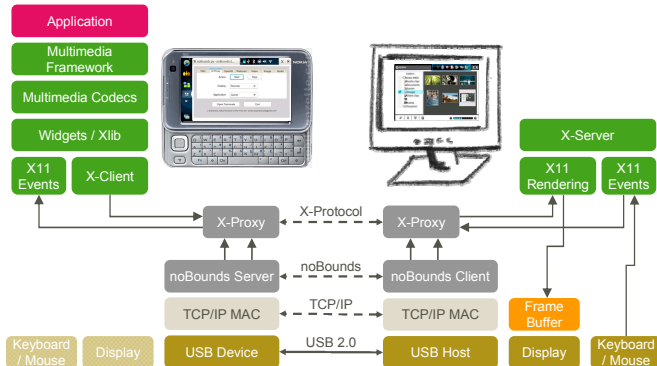


Figure 5. noBounds – X-Server Demonstration

We will demonstrate that the end-user can benefit from the availability of a high-resolution display connected to the mobile device, e.g. for reading / editing documents.

The GLX example, shown in Fig. 6, demonstrates that the noBounds concept overcomes potential limitations on the local device (e.g. missing OpenGL support). The 3D OpenGL based scene is completely defined on the local device and the description is then transferred to the remote device for final rendering.

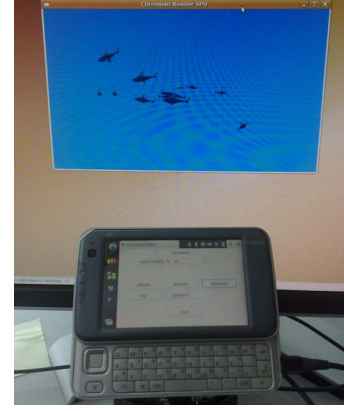


Figure 6. noBounds – OpenGL Demonstration

C. Multimedia Codec

We will demonstrate that streaming of encoded multimedia content enables a bandwidth efficient way to make visual and audio content available at the remote device. The demonstration will show multiple, independent streams, ranging from Web Cam, Pictures, Audio to Video files. In all cases, the remote device is responsible for decoding and final rendering of the data, as shown in Fig. 7. We have used Gstreamer [3] as the underlying multimedia framework.

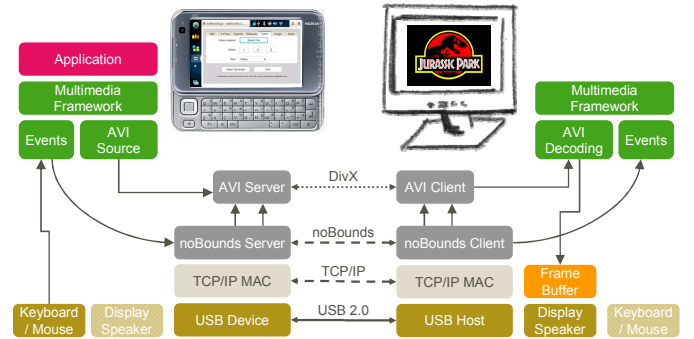


Figure 7. noBounds – Multimedia Codec Demonstration

We will be highlighting the fact that this concept specifically overcomes mobile display limitations and enables unique features by handling different content streams individually.

REFERENCES

- [1] <http://blogs.forum.nokia.com/blog/kate-alholas-forum-nokia-blog/2008/02/12/usb-networking>
- [2] <http://europe.nokia.com/A4568578>
- [3] www.gstreamer.net
- [4] http://en.wikipedia.org/wiki/X_Window_System