GENIVI®

Advanced Driver Information Technology

TECHNOLOGY BRIEF

JULY 2018

Category: Vehicle Domain Interaction

Interactive Cockpit HMI using Surface Sharing

Summary

A holistic digital cockpit HMI with seamless user experience across IVI, Instrument Cluster and other user-interactive displays can be implemented using different approaches:

- Display Sharing
- GPU Sharing (Virtualization)
- Surface sharing
- API Remoting
- Shared State, Independent Rendering

Key Characteristics

- Little to no impact for the content producer (depends on implementation)
- Integration in existing graphics systems is possible with reasonable effort
- The content of individual graphical applications can be shared

Surface sharing creates content once and distributes it within the interactive HMI system to another ECU or virtual machine, where this content is presented.

Description

Modern IVI environments are distributed, cross-platform and multi-display systems. Information is generated from different entities in the system and needs to be shared across the system to where it is best displayed for the user in their current situation. Thus, sharing of graphical content and interaction with it becomes essential. This brief introduces the approach of surface sharing in general and gives a few implementation guidelines.

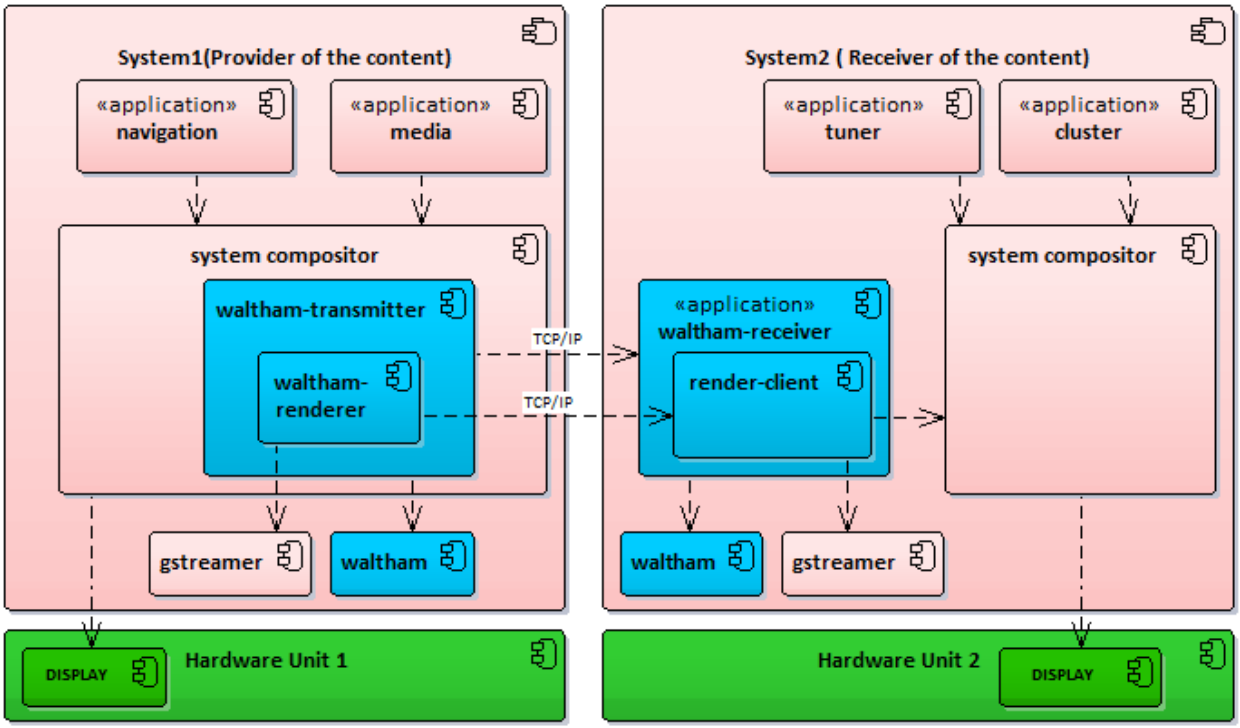
Surface sharing distributes already rendered graphical content, representing the intended graphics from the application, and not expected to be modified on the receiver side. The surface is represented as a two-dimensional image in memory, which can be described with width, height, pixel format and some additional meta-data. Along with the image data, other information, e.g. touch events, can be shared but in terms of size, image data would have by far the biggest share. Therefore, sharing of image data should be the driving point for optimization during the definition and implementation of the sharing mechanisms.

When possible, shared memory between systems should be used. This is available in state-of-the-art virtualized setups, where several systems are running on the same hardware. This feature avoids copying of data. Only some events about the state of the surface sharing are exchanged between the producer and receiver.

On distributed systems without access to common memory all data needs to be shared via network. To reduce the bandwidth usage, video encoding and decoding hardware can be used with reasonable performance.

In addition to the sharing mechanism, a communication protocol is required to request or notify about new available graphical content in the system, forward touch events and control the sharing in general. One implementation of such a protocol is Waltham, which extends the Wayland display server protocol to a networked system. With Waltham, it is possible to define a TCP-IP based protocol that will accomplish the requirements of a concrete system.

A very useful attribute of surface sharing is its ability to be integrated into an existing graphical system. It is important to find a balance between performance, resource consumption and impact to the system.



The diagram illustrates one way to achieve this: On the provider system (left side of the diagram), surface sharing is integrated in the system compositor, which is responsible for compositing the output of the available applications and sending it to the display. The system compositor is a central component and has access to the surfaces from every individual application. It is therefore the right component to provide those surfaces to another system. On the receiver system (right side of the diagram) surface sharing is implemented in an additional application which, from the perspective of the receiver system, is a normal graphical client, like the “tuner” or “cluster” applications. Nevertheless, this application has unique features because it is connected to the producer side of surface sharing and is able to access the shared surfaces. The surface access might be a proprietary implementation, especially in case of virtualized environments. For communication, Waltham can be used but the system designer needs to specify and implement a concrete protocol.

Alternatives & Related Technologies

- Alternatives: Shared State Independent Rendering, API Remoting
- Related: GPU Sharing, Display Sharing, Wayland project, Waltham project

References & Additional Reading

- GENIVI Domain Interaction Project on Graphics Sharing and Distributed HMI
  (https://at.projects.genivi.org/wiki/x/p4T0)

Authors

Eugen Friedrich (efriedrich@de.adit-jv.com)