Analysis of draft tech brief Surface Sharing (temporary page)

`Looking at the document structure & the content of each section.`

> *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 be displayed where it is best for the user in the current situation. Thus sharing of graphical content and interaction with it becomes essentially. This tech brief describes the approach of surface sharing in general and tries to give some guide lines for realize this objective.*

=
(intro)

> *The idea is simple: generate the graphical content in a place where the information is available and share ready rendered content to a place where it needs to be presented to the user. Not so simple and at the same time very important is the concrete implementation of the sharing mechanism. But before jumping into this topic we need to understand what we need to share.*

=
(what to do - generate, transfer)

> *Surface sharing suggest to share already rendered graphical content, with simple words it is a two dimensional image in the memory which represents the current state of some application and can be described with width, height, pixel format and some additional attributes: in one-word meta-data. This means that the receiver of this content will not have the precise description of the content inside of the image and will not be able to modify it partially, this is one of the biggest limitations of this approach. 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 the driving point during the definition and implementation of the sharing mechanisms.*

=
(sharing bitmaps)
(limitations)
(briefly about touch events)

> *We can distinguish mainly between two types of the systems: systems with an access to the common memory and completed separated systems which are connected via some kind of network.*

=
(over network or virtual/shared-mem)

> *A typical example of the systems with an access to the common memory are virtualized systems running on the same hardware. Sharing mechanism should be implemented without any copy operation, only meta-data which describes the graphical content (image in a memory) should be shared. State of the art virtualized setups usually provide common graphical memory which can be shared between virtualized systems.*

=
(shared mem, no copy)

> *On separate systems the graphical content (image data) need to be shared via an available network link. To reduce the network bandwidth usage video decoding encoding and decoding hardware should be used to achieve reasonable performance.*

=
(use network)
(video encoding)

> *Besides the sharing mechanism itself also a communication protocol is required to e.g. request or notify about new available graphical content in the system, forward touch events and control of the sharing in general. One approach to implement such a protocol is Waltham. With Waltham it is possible to define a TCP-IP based protocol that will accomplish the requirements of the concrete system.*

=
(responsibility of protocol)
(waltham is example)

> *Another very important aspect of surface sharing is the possibility of integration in an existing graphical system. It is important to find a balance between performance, resource consumption and impact to the system. Diagram below suggest one approach to achieve this: On the provider system (left side of the diagram) Surface Sharing is integrated in the system compositor which is responsible to perform a composition of all available applications and provide this to the display. System compositor has access to all graphical surfaces from every individual application and therefore it is a right component to be a producer of the Shared Surfaces. On the receiver system (right side of the diagram) Surface Sharing is implemented in additional application. This application will be a new graphical client in the System2 and will provide its graphical content in the same way as "tuner" and "cluster" applications. Nevertheless, this new application is a very special once because it is connected to the producer side of the Surface Sharing and is able to access the Shared Surface which can be very 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.*

=
(integration into existing system - into existing compositor)
(diagram)
(transparent proxy principle - act as client+server to the server+client)
(security-proprietary)
(need to specify protocol in Waltham)

> *There is a different related approach to the Surface Sharing: Display Sharing. Indeed, they are similar. By looking to the diagram above the system compositor will provide a content to the Display and technically it is also a graphical surface. If we will share this display surface it can be also called as Surface Sharing but important point is that we are sharing the entire result from the system compositor and in this case and cannot extract a representation from a single application which is quite a big limitation. Also compared to Surface Sharing there are other possibilities to implement this approach. It can be implemented in the lower levels of the graphics stack, e.g. kernel driver and keep the system compositor unmodified.*

(Display sharing - relationship)

> *Alternatives & Related Technologies*
> *• Alternatives: Shared State Independent Rendering, API remoting*
> *• Related: GPU Sharing, Display Sharing*
>
> *References & Additional Reading*
> *• GENIVI Domain Interaction Project on Graphics Sharing and Distributed HMI*
> *Autors*
> *Eugen Friedrich (efriedrich@de.adit-jv.com)*

`