

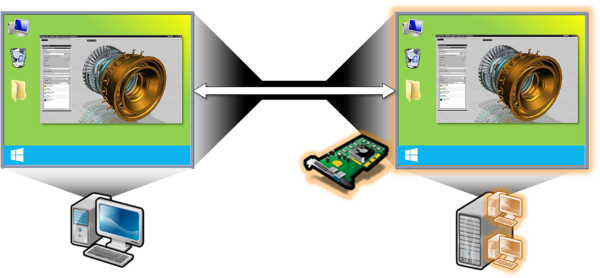
Whitepaper

**Remoting Graphics and Multimedia**

by Dr. Bernhard Tritsch, Ruben Spruijt and Shawn Bass

# 3D Graphics for Virtual Desktop Concepts

This chapter explains the different solutions to deliver 3D graphics for virtual desktops.



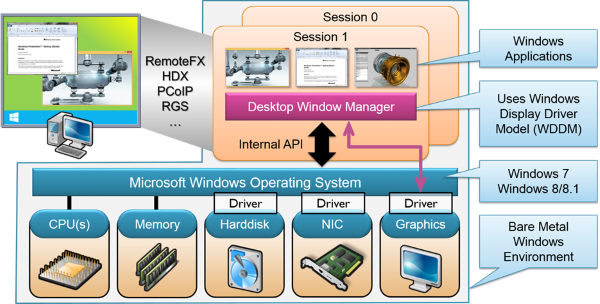
## GPU and APU

Delivering high-end graphics for virtual desktops can be accelerated by using physical Graphics Processing Units (GPUs). A GPU is a highly integrated electronic circuit used in personal computers, mobile phones, embedded systems, graphics workstations, game consoles and cloud platforms. It is designed to perform data manipulation at high performance and in parallel, with the intention to create digital images in a frame buffer and send the results to one or multiple displays. In contrast to a computer’s Central Processing Unit (CPU) with only few cores designed for sequential serial processing, a GPU consists of thousands of smaller cores optimized for dealing with many tasks simultaneously. In a graphics workstation or a cloud server optimized for graphics remoting, GPUs are typically present on a separate video card.

An Accelerated Processing Unit (APU, also Advanced Processing Unit) combines a CPU with additional processing capability designed to accelerate certain types of computations. Such an APU may include a GPU used for general-purpose computing (GPGPU) and for graphics acceleration. In essence, an APU combines a CPU and a GPU on the same die, allowing for high data transfer rates between the two while keeping power consumption at a relatively low level. A good example of a state-of-the-art APU is Intel’s Haswell CPU with integrated Iris Pro GPU. While a dedicated GPU on a separate video card may outperform an APU in many high-end graphics scenarios, lower overall power consumption and lower price are typically advantages on the APU side.

## Bare Metal Graphics Model

This model represents the (blade) PC/workstation remoting architecture where no virtualization is being used. The end user connects via a connection broker to the workstation via remote protocols such as Citrix HDX, Microsoft RDP, VMware Horizon View PCoIP, HP RGS or via Teradici hardware based PCoIP. HP Moonshot is an example of a solution to deliver a virtual desktop running on a bare metal PC-like hardware.



Benefits:

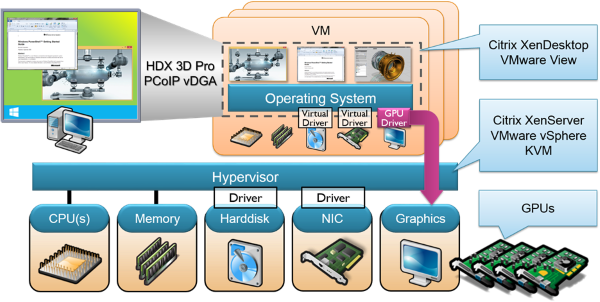
* High-end 3D Graphics and workstation class performance
* Supporting all graphics formats, such as OpenGL, OpenCL, NVIDIA CUDA and DirectX 9/10/11/12
* Full application support, native GPU drivers are being used

Challenges:

* Hardware dependent
* Less cost effective
* No Virtualization benefits such as hardware independence, resource sharing, snapshots, Live Migrations etc.

## Pass-Through or Direct Attached GPU for Virtual Desktops - VDI

In this case, the virtual machine (VM) has a 1:1 relation with a GPU, meaning that the Hypervisor allows the VM full and direct access to the GPU. The native NVIDIA/AMD/Intel graphics driver is installed in the Virtual Machine. VMware call this technology “vDGA - Direct Graphics Acceleration”, Citrix calls this technology “GPU pass-through”. Independent of its name the solution offers full GPU performance and high-end graphics needed for designers and engineers.



Citrix XenServer 6.x and VMware vSphere 5.1 or higher support the pass-through or direct attached GPU for Virtual Desktops.

Benefits:

* High-end 3D Graphics and workstation class performance for virtual desktops.
* Support for all technologies such as OpenGL, OpenCL, NVIDIA CUDA, DirectX 9/10/11/12
* Full application support, native GPU drivers are used.
* Virtual machines with and without direct access to the GPU can be hosted on the platform.

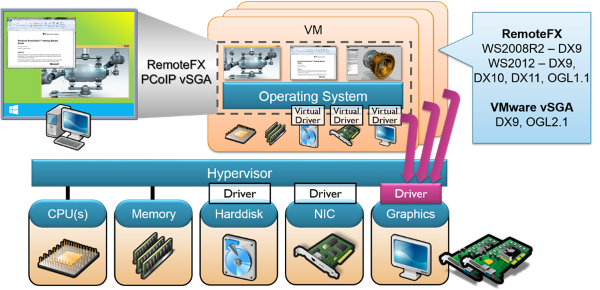
Challenges:

* Live migration of VM with pass-through devices is not supported
* Hardware dependent.

## GPU Sharing for Virtual Desktops, API Intercept

GPU sharing for virtual desktops is a 3D graphics solution suitable for both VMware Horizon View (VDI) and Microsoft Remote Desktop Services. The shared GPU model is also referred to as API intercept where the GPU is managed and owned by Microsoft Hyper-V with RemoteFX or VMware vSphere with Horizon View vSGA (Shared Graphics Acceleration). The graphics API requests generated from the VMs are intercepted via the capture driver in the virtual machine. Typically, this is the synthetic graphics driver included in the integration components of the hypervisor. The graphics requests generated in the VM are redirected to the hypervisors and executed with GPU support. After the graphics processing is finished the resulting graphics data is sent back to the VM. The VM has no direct access to the GPU, as with GPU Virtualization for VDI.

The advantage of GPU sharing is its superior scalability, delivering 10s of VMs or sessions per physical GPU which allow knowledge workers and task workers hardware-accelerated access to Line-of-Business applications and Windows desktops.



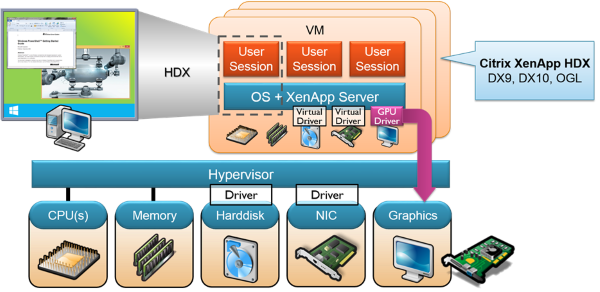
Benefits:

* Cost effective solution for knowledge users and task workers who requires graphic acceleration
* Scalable solution for 10s of VMs or sessions, of course depending on application and usage.
* VMware vSGA implementation can switch between GPU sharing and software 3D renderer. Soft3D is a VMware WDDM driver which is installed in the VM.
* Allows each user to have power user performance with enhanced support for DirectX 3D and Windows Aero
* Supports live migration of virtual machines with VMware vSGA

Challenges:

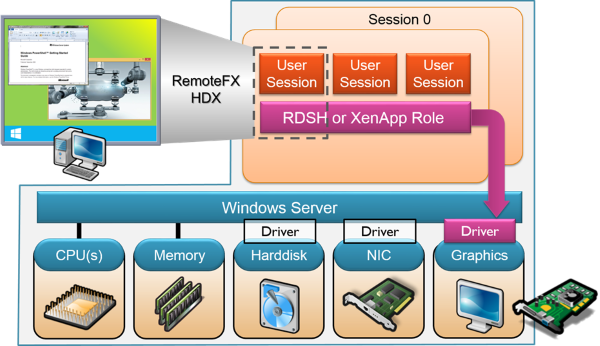
* The solution doesn’t meet the requirements for high-end graphics where users will view, create, manipulate, render complex 2D/3D graphics
* The GPU can become a performance bottleneck as many users will share the same GPU.
* There may be a different set of supported graphics formats for multi-user server platforms (such as Citrix XenApp or Microsoft RDSH), depending on if the server is virtualized or running on bare metal.
* Application compatibility issues due to limitation of 3D APIs supported
  + Limited or missing support of certain OpenGL versions
  + In some cases DirectX support limited to version 9

## GPU Sharing for Virtual Citrix XenApp



## GPU Sharing for Physical Microsoft RDSH and Citrix XenApp

In this model the 3D applications are installed in a multi-user Operating System such as Windows Server 2008R2/2012R2 and published as hosted shared desktop or seamless application. The Citrix XenApp or Microsoft RDSH server need to have access to a GPU, which can be the scenario with bare-metal or a virtual machine with GPU pass-through. It is important to note here that in a pure Microsoft stack, RDSH on top of the Hyper-V hypervisor only supports bare metal GPU access but not PCI device pass-through. RDSH hosted on alternative hypervisors does provide PCI device pass-through access for VMs. In order to leverage a hardware graphics adapter for RDSH, you must enable a GPO setting named “Use the hardware default graphics adapter for all Remote Desktop Services sessions” on Server 2012/2012 R2.



Benefits:

* Cost effective platform to deliver 3D graphics for virtual desktops/apps

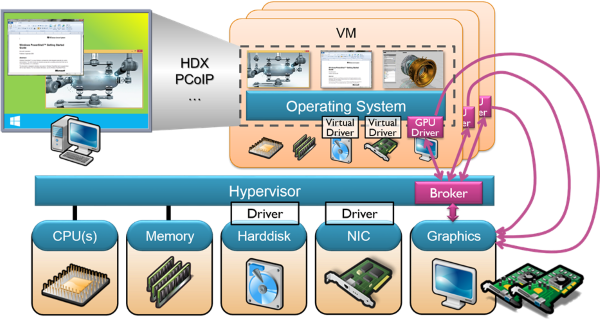
Challenges:

* Some 3D application may not work in a multi-user environment
* The solution doesn’t meet the requirements for high-end graphics where users will view, create, manipulate, render complex 2D/3D graphics
* The GPU can become a performance bottleneck as many users will share the same GPU

## GPU Virtualization for VDI - vGPU

GPU Virtualization for VDI or hardware virtualized GPU offers the benefit of GPU sharing and gives the virtual machine via de native graphic driver access to the GPU. Each Intel or NVIDIA GPU is shared by multiple VMs.

With NVIDIA GRID each VM is configured with a vGPU profile with a configurable static vRAM size and the VM will share GPU processing power. Currently Citrix XenServer 6.2 is the only virtualization solution in the market that supports NVIDIA GRID vGPU. According to a VMware announcement, vSphere and ESX servers will support the vGPU model in early 2015.



Benefits:

* Full support of 3D graphics formats such as OpenGL and DirectX
* Depending on the vGPU profile the solution can deliver high-end graphics performance combined with highest-user density for knowledge workers or operators

Challenges:

* Lower overall VM density per GPU as compared to GPU Sharing for Virtual Desktops
* Little competition, only NVIDIA and Citrix provide this technology today
* No OpenCL and CUDA API support today

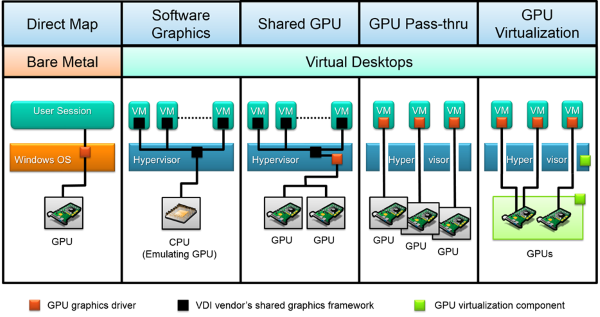
## Application Vendor Support

GPU virtualization and pass-through GPU for VDI allow applications to use vendor-specific graphics drivers. ISVs such as Autodesk, Solidworks and PTC certify their solutions for particular graphics drivers.

* Application certification for NVIDIA GRID can be found here: <http://www.nvidia.com/content/cloud-computing/pdf/GRID_Certifications.pdf>.
* Application certification for AMD FirePro can be found here: <http://support.amd.com/en-us/download/workstation/certified>.

## 3D Graphics for Virtual Desktop Concepts Summary

The five different concepts to leverage 3D graphics for virtual desktops are as follows:



## How to Choose the Right 3D Graphics for Virtual Desktop Solution?

What is the right 3D graphics for virtual desktop solution and how do you make the choice for that solution? First of all, when you don’t know the requirements from end-user, application or IT perspective the easiest and best way to determine the right solution is flipping the coin.

It is key to classify application usage, perceived performance as well as IT and user acceptance. The diagram below is intended to help you when selecting the right model for you as it highlights the different 3D graphics concepts and its characteristics.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 3D Graphics for  Virtual Desktops | High  performance | 3D API Support (OpenGL, DirectX) | Cost effective | Application compatibility |
| Software 3D Graphics | ★ | ★ | ★★ | ★ |
| Bare Metal OS | ★★★★ | ★★★★ | ★ | ★★★★ |
| Pass-through GPU | ★★★★ | ★★★★ | ★ | ★★★★ |
| GPU Virtualisation - vGPU | ★★★ | ★★★ | ★★★ | ★★★ |
| GPU Sharing for VDI | ★★ | ★★ | ★★★★ | ★★ |
| GPU Sharing for XenApp/RDSH | ★★ | ★★ | ★★★★ | ★★ |

## Platforms Accessing 3D Graphics

In today’s world of consumerization of IT, Bring Your Own and Corporate Owned Personally Enabled (COPE) scenarios there is not one single platform anymore. Today’s reality is laptops, hybrid laptop/tablet devices, ultrabooks, zero clients, thin clients, desktops and workstations running Windows, OSX, Android or ChromeOS. On the mobile side there are Android, iOS, Windows Phone, Blackberry and MozillaOS as native platforms and HTML5 as a more generic web application delivery platform.

What is the best platform for accessing 3D graphics applications and/or desktops? Is it a Google ChromeBook with HTML5? Is it a zero client with smartcard integration? Is it a Windows 8 tablet with touch screen? Is it a MacBook Pro accessing AutoDesk Revit? Or is it a Windows 7 corporate laptop accessing PTC Creo? The simple answer is that the best platform for accessing 3D graphics applications and/or desktops does not exist. It all depends on perceived performance, user experience, required functionality such as USB support, multi-monitor support, client printing, network scenarios and many others. So the main question is if you know what your business consumers, your partners and co-workers expect from the solution and if you know the impact on endpoint choice and platform accessing 3D graphics applications.

## Guest OS for 3D Graphics

The guest operating system is the platform being used for delivering Windows or Linux 3D graphics applications. This OS is installed inside the virtual machine, physical server or professional workstation and is responsible for running the 3D graphic application.

What is the best guest OS for delivering 3D graphics applications? The main question to answer is what are the application requirements? Does the application requires a Linux or Windows Client OS, can the application leverage x64 bits? Is VDI or RDSH being used? Do you want to deliver the Windows application as a service through a service provider and what is the impact on licensing? In such as case a Windows Server OS used in a personal mode can be a good solution.

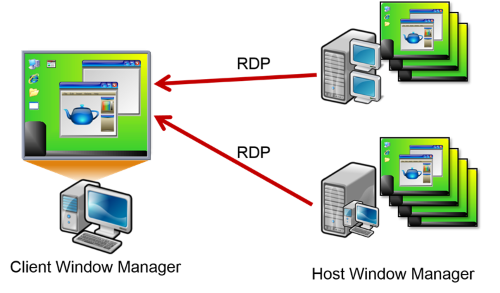
Keep in mind that the majority of 3D graphics for virtual desktop solutions are focused on delivering Windows applications and desktops to users. The VDI products offered by Citrix, Microsoft, NVIDIA and VMware are examples. Remoting protocols and corresponding solutions such as HP RGS and Nice DCV do support both Windows and Linux guest operating systems.

# Remoting Protocols Turned Inside Out

This chapter highlights the concepts behind remote access to contents and display information that are based upon different common graphics formats.

## Graphics Remoting Fundamentals

User experience remoting has been around since the 60s of the last century, initially focusing on text-only remoting across serial lines. Modern graphics remoting goes far beyond such a basic model. It allows rich Windows applications and their graphical output device to be separated by a remoting boundary. It facilitates user interaction with a remote computer system by using a remoting protocol to transfer graphics display data from a host system to the user sitting in front of a client system. User input is transported from the client to the host and replayed there.



Modern versions of remoting protocols, such as Microsoft RemoteFX, Citrix HDX or Teradici PCoIP try to improve performance by taking advantage of physical Graphics Processing Units (GPUs) on the remoting host. In addition, the remoting protocols identify the client capabilities at connection time and constantly analyze the network conditions throughout the entire remoting session time in order to adjust communication settings dynamically. This all helps to make the desktop remoting environment self-adaptable and provide the best performance possible.

When looking at the impact network conditions have on the performance of remoting protocols, most people believe that bandwidth is the most significant factor. While this is true for networks with a low bandwidth profile, it is different for networks with more than 2Mbps per remote user session. In such cases, latency and packet loss become the limiting factors. User experience will typically degrade when latency is more than 50ms and it will be very challenging when latency exceeds the 200ms threshold. Packet loss should be below 1% for a good user experience. However, there are new remoting protocol variants that were specifically designed to perform well in low bandwidth and high latency/packet loss scenarios.

NOTE: Remoting protocols typically don’t limit themselves with respect to available network resources. In other words, if one remote user session requires high bandwidth for rich multimedia content, it may well consume up to 100Mbps if such bandwidth is available.

## Remoting Protocol Features

Remoting protocols run on top of the Internet Protocol (IP), using Transmission Control Protocol (TCP), User Datagram Protocol (UDP) or a combination a TCP and UDP for different aspects of remoting. While older remoting protocols only used TCP, the modern ones use UDP for the graphics remoting aspect.

TCP is a connection-oriented protocol providing high reliability through error checking, congestion control and a built-in mechanism that rearranges data packets in the order specified. It also guarantees that all data remains intact in the packets transferred. But all this makes TCP relatively heavy-weight, significantly reducing graphics remoting performance on low bandwidth and high latency/packet loss networks.

UDP is a connectionless protocol, allowing a program to send individual data packets to another program. There is no error checking and no guarantee that all packets were delivered in the right order. Only if a data packet arrives at its destination it is checked for integrity. This makes UDP light-weight and much faster than TCP, but at the risk of losing chunks of data. The receiver program needs to be prepared to deal with this kind of data loss.

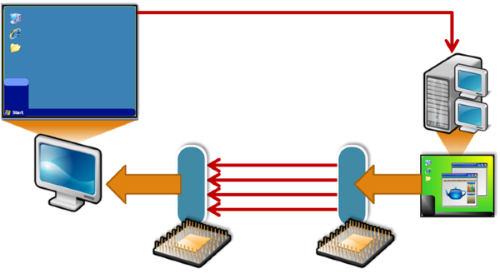
But there is more in a remoting protocol, in particular when it comes to extensibility. The concept of virtual channels provides a way to establish separate streams of data communication while taking advantage of the remote session communication already established. Many remoting protocols use virtual channels to add functions that allow a strict separation from the core features or are not yet specified in the protocol. They represent a platform that future developments can be based on without having to modify the communication methods between host and clients. Examples for virtual channel use cases are joint client and server clipboards or redirecting print jobs to local client printers.

Other notable remoting protocol features include bi-directional audio transmission, client side caching mechanisms, session reconnect after a temporary loss of connectivity, device redirection, multi-monitor support, location awareness and support of Unified Communications. They all are relevant for an acceptable user experience.

## Client Side Rendering versus Host Side Rendering

In a graphics remoting environment, the Windows desktop including its applications is rendered in a different way compared to traditional, local PCs. But what exactly is “rendering”? The most common definition is as follows: If an image described in an abstract model using vector graphics and bitmap primitives is converted into a raster image for output on a screen the term rendering is used. In graphics remoting environments, rendering may happen either on the client side or on the host side.

When graphics primitives are transmitted from the host to the client and rendered on the receiver side, this is referred to as client side rendering. When multimedia data streams are involved, the preferred term is multimedia redirection (MMR). The advantage of this remoting method is that it can greatly reduce host side CPU impact. In addition, redirecting media typically works well in constrained network and due to the nature of the transmitted objects it has a lower bandwidth profile than host side rendering.

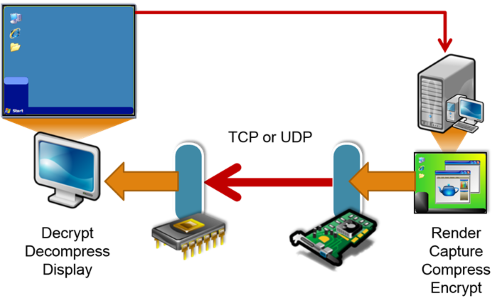


But client side rendering also has many disadvantages: The breadth of features can be very dependent of the client operating system, some forms of media may fall back to server side rendering, client rendered media may require some buffering time, and client side resource requirements may be significantly increased. In addition to that, client side rendering introduces audio/video codec dependencies and possible security vulnerabilities through outdated client side components.

Another significant challenge of client side rendering is the synchronized playback of audio and video in film or animation sequences. This is also referred to as “lip sync”, which is a common term for matching lip movements with spoken vocals. Dedicated codecs and processing pipelines for the separate audio and video data streams in combination with network delays may result in A/V sequences that are out of sync.

Until recently, client side rendering was regarded as the preferred Windows desktop remoting method, providing superior performance and user experience. But modern Windows desktops and applications may use various graphics formats which makes it hard to implement associated rendering algorithms across all popular client platforms. As a result, modern remoting protocols are typically not based on client side rendering anymore.

When graphics primitives are rendered on the host device instead of on the client device, this is called host side rendering. As a result, all graphics types can be by transmitted as a highly compressed bitmap images to the endpoint device in an adaptive manner. Ideally, the host computer takes advantage of one or multiple physical GPUs to improve graphics performance.



The advantage of host side rendering is that it generally is client OS independent and also independent of client hardware. More advantages are that any form of mixed media content can be rendered, media content playback begins immediately, only fairly low client side resources are required and audio/video playback is provided regardless of client side codecs.

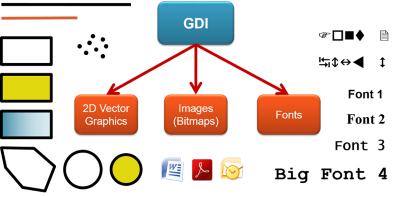
Disadvantages are that host side rendering may generate a very high host CPU impact, it can have A/V sync issues in constrained networks (depending on the protocol’s implementation) and it typically requires a higher bandwidth profile if compared to client or hybrid approach.

As mentioned earlier, Windows desktops and applications are not only built on top of one single graphics format. As a matter of fact, a typical modern Windows system uses multiple graphics formats side by side for composing the desktop. It is important to understand nature and purpose of the individual graphics formats as some of them are treated differently in the different remoting protocols. The following paragraphs introduce the most important graphics and multimedia formats used for Windows desktop composition.

## GDI Remoting

Graphics Device Interface (GDI) is an application programming interface (API) that was developed by Microsoft in the 80s and 90s of the last century. It is responsible for defining and rendering graphical objects on Microsoft Windows. GDI rendering or rasterization is the task of taking an image described through vector shapes or primitives and converting it into a raster image consisting of pixels or dots for output on a GDI-compatible video display or printer. Such a rasterized image can also be stored in a standard bitmap file format.

Typical GDI primitives are lines, rectangles, ellipses, arcs, Bezier splines, filled areas, bitmaps and fonts. A significant capability of GDI is its indirect method of accessing the underlying hardware by drawing on a Device Context (DC). A DC represents an abstraction layer that can be mapped to multiple physical target devices, such as screens and printers. A GDI applications sends its graphics output to the DC and then the Windows system sends the DC content to the target device.



It is important to note that GDI was designed in such a way that it can be hardware accelerated in a graphics card. Every modern PC graphics card supports the full GDI function set in hardware. During the Windows initialization phase, GDI hardware acceleration is registered in the system, significantly reducing CPU load generated by graphics output during system runtime.

In GDI there is no mechanism of synchronizing the DC with the graphics card frame buffer. As a result, GDI is not good at animating graphics primitives. There is no built-in method for double buffering animated output. In addition to that, GDI is a 2D graphics system, so it lacks the capability of rendering 3D primitives. The only 3D capability of GDI is the z-order of a window indicating its position in a stack of potentially overlapping windows. A window in the z-order list overlaps all other windows that are closer to the bottom of the z-order.

Most traditional Windows applications are built on top of GDI. The application windows and dialog boxes consist of basic GDI window objects, such as borders, title bars, captions, control boxes, scroll bars, menu bars and icons. These window objects are built of even more elementary GDI primitives, creating a hierarchical system of GDI primitives and window objects. A message pipeline allows an application to send graphical output to the client area of one of its windows or dialog boxes.

When Windows XP was introduced, GDI was extended by the GDI+ subsystem. GDI+ added more 2D graphics features and the support of modern graphics file formats. The Microsoft .NET Framework includes a managed GDI+ interface through the System.Drawing namespace. Substantial portions of GDI+ are not hardware accelerated.

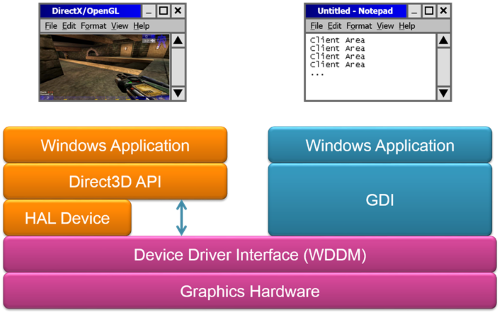
The first designs of the Microsoft Remote Desktop Protocol (RDP) adopted many concepts of GDI and GDI+. The general idea was that remoting client and host negotiated their common set of GDI capabilities during connection, allowing for an advanced level of client side rendering. If the resulting session properties did not include the support of GDI elements like bitmaps, pointers, fonts, brushes, glyphs and standard windows, the communication fell back to pixels. This was referred to as “screen scraping”, where all pixels of a rendered desktop were “scraped” from the host video memory and then transmitted to the client.

Today, both GDI and GDI+ are not as relevant as they used to be in the Windows XP times. They were replaced by more modern graphics formats, such as DirectX and Windows Presentation Foundation. As GDI falls in popularity, so does the relevance of client side rendering.

## DirectX Remoting

DirectX is a collection of application programming interfaces (APIs), covering different aspects of multimedia formats. These include Direct3D (D3D) for 3D graphics, DirectDraw (D2D) for 2D graphics and audio/video APIs called DirectMusic, DirectPlay and DirectSound. DirectX was introduced to compensate for the shortcomings of the Graphics Device Interface (GDI), in particular in Windows games. It is directly connected to display drivers and gets better results at rendering than GDI.

When Windows applications are using Direct3D they write to a 3D surface. In Windows Vista and later versions, Direct3D uses the Windows Display Driver Model (WDDM) to share the 3D surface with the Desktop Window Manager (DWM). DWM then uses this surface directly and maps it on to the window desktop. One of the central element of the WDDM graphics architecture is the new video memory manager. It supports the virtualization of graphics hardware for a range of applications and services like the Desktop Window Manager.



If an adequate graphics card is available, Direct3D can use it for hardware acceleration of the entire 3D rendering pipeline or part of it. The advanced graphics capabilities of the underlying 3D graphics hardware are exposed through Direct3D. Combining such capabilities with the other DirectX technologies allows for advanced multimedia scenarios.

Over the years, Direct3D became a de-facto standard for many software vendors when developing software applications for visualization, games and other high-end graphics tasks. Direct3D 9, Direct3D 10 and Direct3D 11 are only available on Windows Vista and later due to the fact that they require the Windows Display Driver Model.

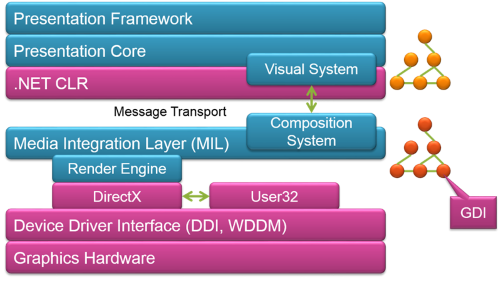
NOTE: Microsoft released Direct3D in 1995, which became a competitor of OpenGL. In December 1997, Microsoft and Silicon Graphics initiated a joint effort with the goal of unifying the OpenGL and Direct3D interfaces.

Remoting DirectX is typically done on the host side, resulting in very poor performance when using the older remoting protocols. However, there are forms of client-side DX remoting that are enabled by both Microsoft (Aero Glass Remoting in RDP) as well as Citrix XenDesktop (Desktop Composition Redirection). These forms of DX remoting even work well in decent WAN conditions, but they are designed to support basic DX remoting and are not designed to support high-end DirectX graphics remoting. By redirecting DirectX calls to the client device, you can reduce the server load on the hosted desktop platform thereby scaling more users. The downside of this approach is a more limited set of client hardware / OS support (though fallback to server side rendering occurs in those cases).

## WPF Remoting

Windows Presentation Foundation (WPF) was introduced with Windows Vista as part of the .NET Framework 3.0. It was designed to be the successor of GDI for the standard graphical user interface of Windows desktops and applications.

WPF is completely built on top of DirectX, enabling modern UI features like transparency, gradients and transforms. One of the major WPF design goals was to provide a consistent programming model for building applications. In particular, it separates the user interface from business logic by providing a markup language (XAML) to define UI elements and relationships with other UI elements.



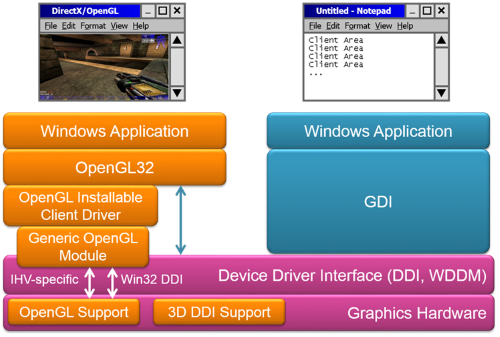
WPF applications allow for a variety of runtime options. They can either be deployed as stand-alone desktop programs or hosted as an embedded object in a website. Independently of the deployment method, WPF supports a basic set of common user interface elements for 2D and 3D rendering, including vector graphics, runtime animation, and pre-rendered media. These elements can then be linked and manipulated based on various events, user interactions, and data bindings. One such element is the 2D surface that is generated by all open GDI applications. This GDI surface is mapped on the WPF-based Desktop Window Manager. As a result this concept for desktop composition allows to combine 2D and 3D elements seamlessly, including moving bitmaps, transparency and anti-aliasing. One of the biggest advantages for GDI applications when used in the WPF context is that the application logic doesn’t need to re-render graphical content when the application window comes to the foreground after it was covered by another window before.

Remoting WPF is identical to remoting DirectX, which means that it is typically rendered on the host side and can be GPU accelerated.

## OpenGL/WebGL Remoting

Open Graphics Library (OpenGL) is a multi-platform application programming interface (API) widely used in 2D and 3D applications. It was developed by Silicon Graphics Inc. in the 90s and is managed by a non-profit technology consortium, the Khronos Group, since 2006. The OpenGL API consists of over 250 different function calls to draw complex three-dimensional scenes from simple primitives. It is capable of interacting with a physical GPU for hardware-accelerated graphics rendering. Version 1.0 of OpenGL was released in January 1992. New versions incorporate a number of extensions, defining a new set of features which all conforming graphics cards must support. OpenGL 1.1 was released in 1997 and is the supported version in Remote Desktop Protocol (RDP) 8.1 including RemoteFX. OpenGL 1.2, 1.3, 1.4 and 1.5 followed in 1998, 2001, 2002 and 2003 respectively. OpenGL 2.0 introduced a Shading Language in 2004 and was succeeded by version 2.1 in 2006. OpenGL 2.1 is the supported version in VMware vSGA.

OpenGL 3.0 was released in 2008 and it introduced a deprecation mechanism for API revisions. OpenGL 3.1, 3.2 and 3.3 followed in the years until 2010. HDX 3D Pro is fully compatible to OpenGL 3.3 and earlier. The design goal of OpenGL 4.0 was to support DirectX 11 compatible hardware. It was also released in 2010 alongside version 3.3. New NVIDIA and AMD graphics cards led to versions 4.1, 4.2, 4.3 and finally 4.4. In general, OpenGL releases are backwards compatible. Most graphics cards released after the release date of a particular OpenGL version support those version features, and all earlier features.



In remoting scenarios, OpenGL is typically rendered on the host side, and it can greatly benefit from GPU acceleration.

WebGL is a cross-platform web standard for a low-level 3D graphics API. It is based on OpenGL ES 2.0 and exposed in Apple Safari, Google Chrome, Mozilla Firefox and Opera browsers through the HTML5 Canvas element as Document Object Model interfaces.

## OpenCL Remoting

Open Computing Language (OpenCL) is an open, cross-platform standard for writing gaming, entertainment, scientific and medical software. It was designed for parallel programming of CPUs, GPUs, digital signal processors and field-programmable gate arrays while it interoperates with OpenGL and other graphics APIs. Like OpenGL, OpenCL is maintained by the non-profit technology consortium Khronos Group. It has been adopted by vendors such as NVIDIA, AMD, Intel, Apple, Qualcomm, Samsung and ARM Holdings.

Version 1.0 of OpenCL was released in December 2008, allowing applications to tap into multi-core CPUs and GPUs. OpenCL 1.1 and 1.2 followed in June 2010 and November 2011 respectively, adding functionality for enhanced parallel programming and improved OpenGL and DirectX interoperability. OpenCL 2.0 is the latest significant version. It was released in July 2013 and is designed to further simplify cross-platform parallel programming.

Remoting OpenCL is pretty much identical to remoting OpenGL. Rendering typically happens on the sender side.

## Flash Remoting

Adobe Flash is a popular multimedia platform to create Web advertisements, interactive games and Rich Internet Applications (RIAs), allowing the combination of still images, animation, vector graphics, audio and video. Viewing and interacting with the interactive multimedia content of a Flash application requires the Adobe Flash Player which supports the Flash Video Format (FLV). From a playback perspective, the Flash Player has a lot in common with a custom-developed video player using a proprietary multimedia format. But in contrast to a video player, Flash allows user interaction through mouse, keyboard, microphone and camera. This includes bidirectional streaming of audio and video data. A clear advantage of Flash is that the Flash Player is available free of charge as a plug-in for all major Web browsers and for many operating systems, smartphones and tablet devices.

Creating interactive Flash applications is based on an object-oriented programming language called ActionScript. Professional development tools are required to build such application. Since a couple of years, the introduction of new or updated Flash applications is declining on websites. However, there are still many Flash applications in use.

Remoting Flash is typically done on the host side, resulting in very poor performance when using the older remoting protocols. Only Citrix and Dell implemented a technology called Flash Redirection which relies on an existing Flash Player on the client side. Flash Redirection improves usability and reduces bandwidth requirements significantly, but also requires a client that is capable of installing and maintaining a Flash Player. There are some negative aspects of performing client side Flash remoting such as a dependency of keeping the server and client Flash versions up to date as well as some security related risks when redirecting Flash to the client system (i.e. a server hosted Flash control could infect a client PC since the Flash content is rendered on the client).

## Silverlight Remoting

Microsoft released Silverlight in 2007 with the ambition to replace Adobe Flash as the most popular application framework for interactive multimedia applications. Now in its fifth version, the Silverlight runtime environment is available as a plug-in for Web browsers running under Windows and Mac OS X. In addition, Silverlight is one of the Windows Phone development platforms, even though the resulting applications can only run natively on Windows Phone but not from Web pages in Internet Explorer on Windows Phone, Windows Mobile or Windows RT. Like with Adobe Flash, Silverlight remoting is done on the host side. There is no technology available today that allows redirecting Silverlight content even if a Silverlight runtime is installed on the client side. As a result, Silverlight performance is degraded when using the older remoting protocols.

## Audio/Video Remoting

Playing back audio and video streams is very easy on Windows systems. The built-in Media Player supports a number of multimedia file formats, such as WAV for audio and WMV or AVI for video. RealVideo and Apple QuickTime can be seen as competitors.

Most video files consist of two or more types of files objects. One is the media container which describes the overall structure while another represents the compression/decompression (codec) algorithm used inside the container.

Here is a short overview of the most popular file formats and codecs:

* MPEG-1 and MPEG-2 files are based on a compression format standardized by the Moving Picture Experts Group.
* MP4 is a video format that enhances the MPEG standard by the support of video/audio "objects", 3D content, low bitrate encoding and Digital Rights Management. In addition, it allows the separation of audio and video tracks. Video is compressed with MPEG-4 and audio uses AAC compression.
* Audio/Video Interleave (AVI) was developed by Microsoft and most commonly contains MPEG or DivX codecs even though it can contain almost any codec.
* The Windows Media Video format (WMV) was also developed by Microsoft. Initially designed for Internet streaming, it is now a common format used for video playback.
* QuickTime was developed by Apple. The format contains one or multiple tracks containing video, audio and other objects.
* The RealMedia format was created by RealNetworks, allowing media streaming of both audio and video data.
* The RealVideo format was developed by Real Media with a low bandwidth use case in mind.

The most popular video codec today is H.264/MPEG-4 Part 10 or AVC (Advanced Video Coding), released in May 2003. It is a block-oriented motion-compensation-based video compression standard, also used as one of the video encoding standards for Blu-ray discs. H.264 can be implemented in hardware which allows modern GPUs to accelerate video compression and decompression on hardware level. By default, H.264 compression is lossy, but it can be configured to be lossless, which is an important aspect for use cases such as medical imaging.

Many remoting protocols try to redirect audio and video data streams to the client if on session connection the required codec packages can be found there. If a video stream is decoded in host video memory and re-encoded for remoting the content, this comes with substantial resource requirements.

NOTE: Remoting protocols implementing host side rendering typically use H.264 to encode and compress the Windows desktop into video frames before sending it down the wire. Taking advantage of the H.264 hardware encoder on the GPU improves remoting performance significantly.

## HTML5 for Remoting

Hypertext Markup Language version 5 (HTML5 for short) is the advanced version of HTML, used for presenting text and multimedia content in the World Wide Web. An important new feature in HTML5 is the way it embeds graphics, audio, video and interactive documents, allowing to create Web applications. New syntactic features provide a simpler way to manipulate multimedia and graphical content without the need to install additional browser plugins. The latest versions of all modern browsers, such as Internet Explorer, Firefox, Chrome, Safari and Opera, support HTML5 to a certain extend. A public HTML5 test site (http://html5test.com/) creates a HTML5 compatibility score.

An interesting aspect of HTML5 is its capability to deliver remote desktops via native browser components. This requires WebSockets to create a TCP connection channel for continuous transmission of data throughout the entire remote session lifespan as opposed to the connectionless communication via HTTP. In addition, a native HTML5 element called Canvas is needed that has the ability to control single pixels discretely and output complete raster images at high speed. In combination, WebSockets and Canvas provide the necessary mechanisms to connect to a remoting host from a browser and render the 2D graphics output dynamically. As a result, the browser receives the remoting data through WebSockets and uses the Canvas element to draw the desktop. A gateway component or driver between the host and the browser is required to render the content received through the remoting protocol and convert it into a binary data stream of desktop frames retransmitted through WebSockets.

Among industry experts and market analysts, remoting clients based on HTML5 are regarded as a top candidate for the preferred future technology. But at this stage it is too early for declaring HTML5 as the successor of native remoting clients.

## CUDA in Graphics Remoting Environments

The Compute Unified Device Architecture (CUDA) is NVIDIA’s parallel computing platform and programming model. CUDA was designed to be used with NVIDIA GPUs, giving developers direct access to instruction set and memory of the graphics chip. Use cases are not only limited to high-end graphics. CUDA GPUs can also be used for general purpose processing, an approach called GPGPU (General Purpose Computation on Graphics Processing). The advantage a GPU has over a CPU is that GPUs are optimized for running a big number of concurrent processes or threads side by side, resulting in superior overall performance. This makes GPGPU extremely powerful for scientific calculations and cryptography.

NVIDIA offers CUDA-accelerated libraries and extensions to standard programming languages such as C and C++, allowing developers to take advantage of the GPU’s capabilities. Vendors like Adobe or Autodesk use such features to accelerate their 3D CAD/CAM programs.

CUDA-enabled applications only run if there is a physical NVIDIA GPU present on the system. Remoting such applications also benefits of the hardware acceleration on the GPU.