Taking Measurements from the End User Perspective

The Challenges with GUI Scripting

Remote Desktops

In remote desktop solutions, the user interacts with a pictorial representation of the real desktop, which resides within the datacenter. The applications that users interact with do not reside locally to the client desktop delivery end point. As such, there are few application objects to interact with from an automation GUI perspective. All the useful GUI application items, such as window objects, child window objects, text, combo boxes, list boxes, radio buttons, etc., are not available at the client's desktop delivery end point.

Client Side GUI Scripting

GUI automation scripting at the client side simply does not work reliably. Some solutions script at the client side and try to synchronize with the application under test using bitmap comparison. This is not a reliable method of implementing synchronization, GUI scripts break frequently, even when the system is unstressed.

There are massive obstacles in building a robust and reliable GUI automation script which can dynamically synchronize with the application as its behavior changes, as load increases, and can handle unexpected dialog boxes or known functional issues. Greater access to the OS and application objects is required for resilient and successful scripting. For example, it is necessary to be able to kill a process by its ID, or to count the total number of open windows on the desktop and the application objects.

Server Side GUI Scripting

Scapa TPP uses server side GUI scripting on Terminal Servers, Microsoft® Remote Desktop Servers or the Hosted Virtual Desktop (HVD) for Virtual Desktop Infrastructure (VDI) solutions.

GUI automation tools, such as Wintask and AutoIT are used to script the applications. These tools interact with the application at the object level, which provides the necessary access to enable robust and reliable scripting.

An advantage of scripting at the server/HVD layer is that the GUI automation script is protocol independent, which means you can Script once and run the same script on Citrix® XenApp™, Citrix XenDesktop™, VMware® View™, Microsoft Remote Desktop and traditional thick clients.

Ensuring the Server and Client are in Sync.

When you script at the server/HVD side, measurements are taken from the server/HVD perspective by default. For instance, the time taken for a user to see the first window when starting Microsoft Excel® is not measured. Instead, you are measuring the time taken for the server to generate the remote desktop GUI.

In order to obtain the measurements from the desktop delivery end point and, therefore, the end user experience, Scapa has implemented ScapaSYNC, which ensures that the script running on the real remote desktop cannot run ahead of the client. Scapa TPP ensures proper synchronization between the end user and the server/HVD side, allowing true measurements to be taken from the end-user perspective.

Virtual Channels

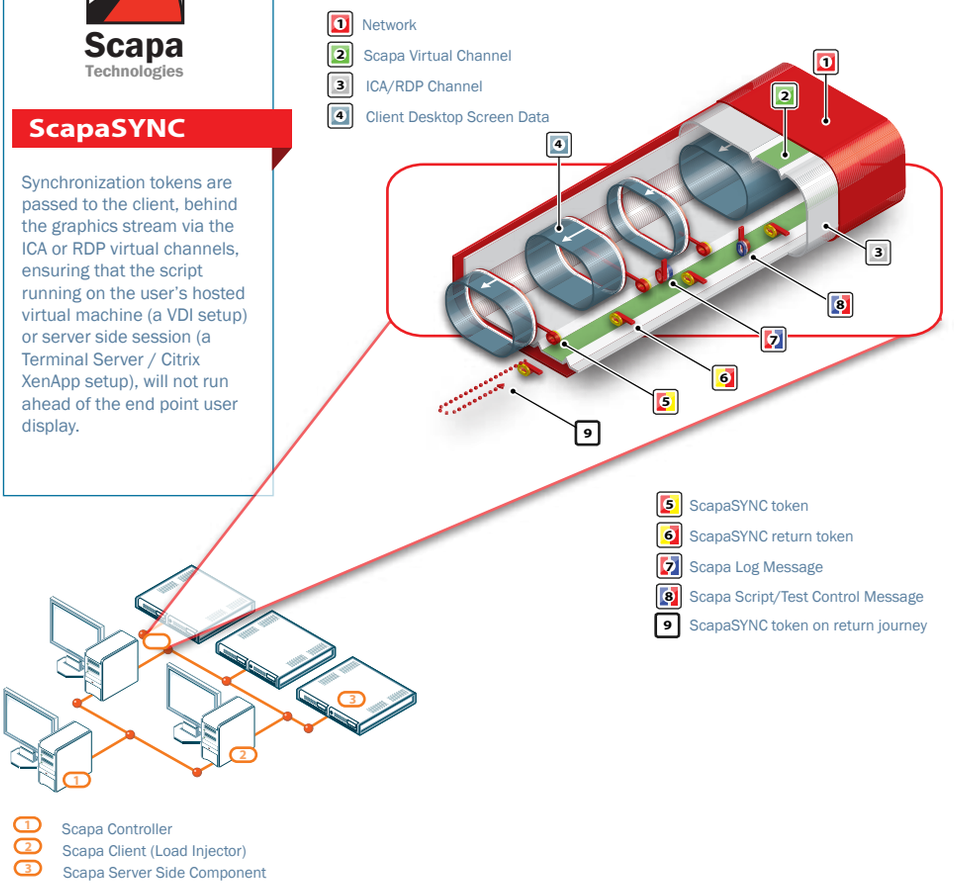
Citrix uses the ICA® protocol, VMware and Microsoft use the RDP protocol. Both of these protocols have a 'main' channel to deliver the pixel and glyph update information to the client delivery end point. This channel has priority over all other channels. Other channels are used for printer and client drive mapping etc. In addition to the default channels, Scapa TPP installs an extra channel. Scapa TPP uses this channel to send synchronization tokens to the client. When a token is sent, the server side GUI automation script is suspended. When the client receives the token an acknowledgement is sent back to the server/HVD and the GUI script continues.

The scripter and the GUI automation script are responsible for dynamically synchronizing with the application objects and the ScapaSYNC tokens ensure that the script synchronizes with the experience of the end user. By placing Scapa Start Timers at the correct position in the script (such as just before starting an application), the next script call will ensure that the application window exists on the server/HVD. This call is followed by a Scapa Synchronization token and then, finally, by the Scapa Stop Timer. As the graphics channel has priority and the channels are serialized, we know that the ScapaSYNC token will be delivered to the client immediately after the pixel/glyph update information. Once the client responds, the script will continue, the Scapa Stop Timer will be called and we will have measured the experience from the end user perspective.

The advantage of this approach is that we can separate the end user experience from the server experience and record the overall latency down the ICA/RDP protocols per user, per location and per server/HVD. This information is invaluable when testing and comparing end user experience in remote office locations.

Client Side Synchronization: How it Works

SCAPA VIRTUAL CHANNEL VISUALIZATION



Scapa Technologies

ScapaSYNC

Synchronization tokens are passed to the client, behind the graphics stream via the ICA or RDP virtual channels, ensuring that the script running on the user's hosted virtual machine (a VDI setup) or server side session (a Terminal Server / Citrix XenApp setup), will not run ahead of the end point user display.

Test Control

In addition, in pre-defining test load behaviors, Scapa TPP has the ability to drive load levels interactively. It is possible to increase and decrease user concurrency at will. The message to run users is sent from the Scapa Controller to the client which then sends a message via the virtual channel to the appropriate user session running the GUI script.

Messaging

Scapa log messages are sent from the server to the client via the virtual channel and then passed to the centralized SQL database.

Log messages are invaluable in understanding problematic systems.

For more information: quotes/case studies/references please contact Scapa : www.scapatech.com | Contact@scapatech.com
Tel. USA : +1 (415) 287 4126 | Tel. UK : +44 (0)131 208 0652

Scapa Technologies