# How to switch from SphinxLib to the CSGenICAM-SDK

*Whitepaper*

F. Sczech

July 2, 2021

## Summary:

This whitepaper will indicate the differences in the usage of the two GenICam-based SDKs. It will point out the differences and new possibilities of the new CSGenICam-SDK Additionally a short introduction into the RealTime-No-packet-loss possibilities of the CSGenICam-SDK will be given.

## Content

## 1. Introduction

The goal of the Chromasens GmbH is to provide a seamless and user-friendly integration of the new camera families into the user environment.
In order to achieve this goal, a new GenICam-based SDK has been developed which will be able to support the new cameras regardless of their interface. For example, GigE-cameras or CoaXPress will be accessible using the same functions.

Due to the architecture of windows based systems, running GigE-cameras often resulted in problems of sporadic packet losses. The new CSGenICam-SDK now provides a solution for these problems.
By providing the possibility to use a real time addon which will use dedicated CPUs those problems can be overcome!

This paper will demonstrate how to switch from the SphinxLib to the new CSGenICam-SDK.

## 2. Switching from Sphinx SDK to CSGeniCam-SDK

Please note that the code examples are not complete. Declarations of variables and proper error handling is missing here. For the complete implementation you should refer to the delivered examples!

### 2.1 Initialization and Discovery

**[SphinxLib]**:
No Initialization needed

```
  // discovery devices
  error = GEVDiscovery(&dis,NULL,200,0);
```

**[CSGenICam]**:
Initialize the library first before performing the discovery process.

```
// Initializing the SDK
status = csiInit(CSI::csiLogLevel::CSI_LOGLEVEL_TRACE);
// start device discovery
result = csiDiscoverDevices(&discovery, 200, progressCB);
```

### 2.2 Open a camera

**[SphinxLib]**:

```
  // init GigE device
  error = GEVInit(camera, &con, error_callback_func, 0,EXCLUSIVE_ACCESS);
  // init xml parser
  error = GEVInitXml(camera);
```

[CSGenICam]:
```
status = csiOpenDevice(choosenDev.deviceIdentifier, &dev, 200,
       csiDeviceAccessMode::CSI_DEV_MODE_EXCLUSIVE);
```

## 2.3 Getting and setting camera features

[SphinxLib]:

Get a feature
```
  error = GEVGetFeatureInteger (camera,(char *)"Height ",&height);
```
Set a feature
```
  error = GEVSetFeatureInteger(camera,(char *)"Height ", height);
```

[CSGenICam]:

Get a feature
```
error = csiSetFeatureInt(choosenDev, "Height", height)
```
Set a feature
```
error = csiSetFeatureInt(choosenDev, "Height", &height)
```

## 2.4 Retrieving images from the camera

[SphinxLib]:
```
// open stream channel
error = GEVOpenStreamChannel(camera, con.AdapterIP, con.PortData,0 );
// start acquisition
error = GEVAcquisitionStart(camera, acquisition_counter);
// retrieve image from camera
error = GEVGetImageBuffer(camera,&img_header, ppixel[0]);
// stop acquisition
error = GEVAcquisitionStop(camera);
```

[CSGenICam]:
```
// Create a single data stream with five buffers
result = csiCreateDataStream(choosenDev, 0, &stream, 5);
// Register new image event
result = csiRegisterEvent(stream, CSI_EVT_NEWIMAGEDATA, &newBufferEvent);
// Create an event fetcher where the new image data event will be handled
EventFetcher fetcher(newBufferEvent);
// start acquisition
result = csiStartAcquisition(choosenDev, CSI_ACQUISITION_CONTINUOUS);
// Wait or do whatever is needed……
……
// stop acquisition
csiStopAcquisition(choosenDev);
```

## 3. Using the Real Time functionality of the CSGeniCam-SDK

The good news is, that you do not need to change anything in your code in order to use this functionality.

The only difference will be that you need to install the option for the real time GenTL and reserve 1 or 2 CPU cores for the extension.

You will also need a dongle in order to use this possibility.

Afterwards the camera will be found automatically by using the discovery as it was used before. The only difference will be, that a different transport layer will be used.

## 4. Conclusion

As shown in  chapter 2 the changes for the standard use cases are minimal.

To switch from the SphinxLib to the new Chromasens CSGenICam-SDK will not cost a significant effort.

The used functions are very similar to each other and are of course based very closely to the GenICam-standard.

The CSGenICam-SDK is delivered with samples which demonstrate how to use the functions and to set up your program in order to retrieve images and handle the available camera features.

Additionally, convenience functions like e.g., file download, reference generation or similar are provided.

Using Real time or the standard approach will not result in any changes of your software when using the CSGenICam-SDK!

The SDK can be downloaded from the Chromasens website by following this link(It is integrated in the installer package for the GCT-tool):

https://www.chromasens.de/gct2-current

## References

[Please find examples of previous published White Papers here: https://www.chromasens.de/de/white-paper]

**Chromasens GmbH**
Max-Stromeyer-Straße 116
78467 Constance
Germany

Phone: +49 (0) 7531 876-0
Email: info@chromasens.de