Introduction to TWAIN Direct™ (DRAFT COPY)

December 2, 2014
Revision 0.1

History

Date	Version	Comment
October 27th, 2014	0.01	Initial draft.

Notes

Notes
• (none)http://jsonlint.com/

Contents

Overview

Summary

TWAIN Direct is a command set used by applications to control scanners. Its goal is to replace the existing scanner APIs as the preferred method of image capture control. This document discusses the current challenges involved in capturing image data, and how TWAIN Direct addresses those issues.

Target Audience

A scanning environment consists of the following:

- A scanner (and therefore a scanner vendor).
- An end user, who operates the scanner.
- Application software used by the end user (and therefore an application vendor).
- IT departments to maintain the environment for the application.

TWAIN Direct is primarily focused on application vendors. Applications represent the needs of the end users, and there are considerably more application writers than scanner vendors. Improving the development experience in this area offers the most potential for influencing the market. The expectation is that focusing on this segment will inevitably leads to benefits for the other segments.

Current Challenges for Application Vendors

In today's environment, when a developer decides to add scanning capabilities to their application, they must be prepared to encounter the following issues:

- For historical reasons Image Capture APIs are generally limited to access by the C and C++ programming languages. Third party toolkits are required when interfacing to other popular languages or to web browsers.

- Developers must decide which Image Capture API to use (e.g., TWAIN, WIA, ISIS, SANE, etc). This decision is influenced by the operating system the application runs on and the APIs supported by the scanner. This makes it harder for developers to switch to different scanner products or to support new ones.

- Image Capture APIs are complex, reflecting the native complexity of scanners, which

often support a large number of possible settings (e.g., compression, brightness, contrast, region of interest, deskew, barcode detection, etc). The TWAIN Specification currently details more than 150 capabilities. Scanner vendors may add their own custom capabilities. This complexity often causes developers to seek a minimal solution to their problems, even when the scanner is capable of providing a better user experience.

- Instead of focusing solely on solving the end user's problem, the bulk of development time is devoted to defensive programming dealing with the variable feature set of scanners and the reliability of their drivers. Development of a robust application may take weeks or months.

- Scanner vendors supply drivers for their devices (e.g.: TWAIN, WIA, ISIS, SANE, etc). End users and IT personnel must install these drivers on their systems and upgrade them as needed. In this model applications talk to drivers, and drivers talk to scanners. The communication between a driver and a scanner is proprietary, making it harder for developers to diagnose problems without the aid of the scanner vendor.

- The current Image Capture APIs were developed for peripheral communication protocols (e.g. IEEE 1284, SCSI, USB, etc). Application control of network scanners has not been standardized and adopted across the industry for all operating system platforms.

TWAIN Direct

What is TWAIN Direct?

- It is a command set that allows applications to talk directly to scanners without vendor specific drivers (referred to as Scanning Without Requiring Drivers, or SWORD).

- It is an initiative dedicated to minimizing the coding effort developers need to support fully featured image capture solutions in their applications.

The Goals

- **Support direct communication between applications and scanners:** TWAIN Direct is not tied to any particular wire protocol, however, given the growing interest in mobile devices, the TWAIN Working Group is focusing TWAIN Direct on network connectivity. This network can be through a Cloud or as a direct connection between the user's device and the scanner. This design eliminates the need for vendor specific scanner drivers, reducing the support requirements for both scanner vendors and IT departments.

- **Simplify development:** Target the effort to the need of the application writers. Adding basic image capture support to an application should take hours. Adding more complex support should only take days. The TWAIN Working Group supplies sample code in addition to the TWAIN Direct specification to help developers get started.

- **Support modern programming languages (not just C and C++):** Except for the image data, the command set in TWAIN Direct is composed of JSON data in the form of strings. The command set is human readable. The intention here is to make it easier for application writers who need to work with different programming languages for different mobile platforms.

- **Support the full functionality of scanners:** Seamlessly integrate custom features from one or more vendors within the same TWAIN Direct command set. Custom features are the driving force behind new additions to TWAIN Direct, so the standard makes it easy and safe to use them. The application can recommend certain scanners for the best user experience, while still being able to function with less capable scanners.

- **Provide applications access to all of the metadata associated with an image and seamlessly integrate custom metadata without risk of name space collisions:** The TWAIN Specification provides the starting point for standardized metadata items offered within TWAIN Direct.

- **Describe a baseline format for images that all scanners support, which can be easily expanded to encompass future functional requirements:** A proposal for PDF/raster is currently in-progress. The format will be verified as valid PDF, but constrained in such a way that application writers can extract the image data without having to use a PDF library. The format is also friendly to scanner vendors, who need to stream image data without altering the contents of the image itself.

- **Maintain version independence:** All versions of TWAIN Direct are interoperable across all applications and scanners. The TWAIN Working Group has more than 20 years of experience in this area with the TWAIN specification.

- **Emphasize success**: Scanners must deliver images, unless the application states otherwise. For example, if an application *requests* JPEG compression, but the scanner cannot deliver it, then the scanner delivers the default (uncompressed raster data). If the application *requires* that JPEG compression is used, and the scanner cannot deliver it, then and only then will the scanner return a configuration error.

- **Anticipate the future:** Commonly used ether and Wi-Fi connections are not currently as fast as the best peripheral communication protocols (ex: USB 3.0). This will not always be the case. The expectation is that faster scanners will eventually migrate to this platform, and TWAIN Direct is designed to be ready.

- **To encourage adoption by application writers TWAIN Direct proposes a three-step path:**

- 1.Provide a way for legacy TWAIN scanners to benefit from TWAIN Direct, leveraging off the existing installed base.

  2. Demonstrate a way for existing USB scanners to expose a TWAIN Direct interface without modifying the base hardware (referred to as a sidecar solution). To accomplish this, the scanner is plugged into a small device that implements TWAIN Direct for it. This device could come from the scanner vendor or a third party provider.

- 3. Encourage scanner vendors to natively support TWAIN Direct.

The Non-Goals

TWAIN Direct defers to industry standards and enablers for the following items. While these items are not defined by TWAIN Direct, recommendations for their use will be found within the associated sample code.

- Scanner discovery, registration and authentication.

- Communication security (when data is in motion).

- Image and metadata security (when data is at rest).

The TWAIN Working Group is working with the Google Cloud Print team to address these issues, and looks to work with other interested groups.

A Brief Preview of TWAIN Direct

This section provides a glimpse of the JSON command set TWAIN Direct currently in development by the TWAIN Working Group. This content is not final and may differ substantially from the format in the final specification

Sample Configurations

A configuration is a hierarchy of nest objects used to address a scanner's topology. For instance, a scanner with a flatbed and an automatic document feeder may have a topology that includes flatbed, feeder front, feeder back in combination with pixelFormats of color, grayscale and black-and-white pixel formats. Each object in the hierarchy addresses one item in the topology. The following is an example of scanning from any source (flatbed, feeder, etc) grayscale images that have a resolution of 300dpi.

```
{
    "actions": [
        {
            "action":"configure",
            "streams": [
                {
                    "sources": [
                        {
                            "source":"any",
                            "pixelFormats": [
                                {
                                    "pixelFormat":"gray8",
                                    "attributes": [
                                        {
                                            "attribute":"resolution",
                                            "values": [
                                                { "value":"300" }
                                            ]
                                        }
                                    ]
                                }
                            ]
                        }
                    ]
                }
            ]
        }
    ]
}
```

The above examples produces an image, even if the scanner does not support gray8 or a resolution of 300dpi. If an application requires specific values it can ask for them. The next example requests a color image at 300dpi or 600dpi. The scanner is free to deliver a grayscale or a black and white image, but if it cannot provide 300dpi or 600dpi, then it must fail.

```
{
    "actions": [
        {
            "action":"configure",
            "streams": [
                {
                    "sources": [
                        {
                            "source":"any",
                            "pixelFormats": [
                                {
                                    "pixelFormat":"gray8",
                                    "attributes": [
                                        {
                                            "attribute":"resolution",
                                            “contingency”:”fail”,
                                            "values": [
                                                { "value":"300" }
                                                { "value":"600" }
                                            ]
                                        }
                                    ]
                                }
                            ]
                        }
                    ]
                }
            ]
        }
    ]
}
```

Sample Metadata

This section is not meant to be exhaustive or definitive, it may change in format and content in the final version of the TWAIN Direct specification. It's provided as a guide to the format of the data and the kinds of information that the TWAIN Working Group is investigating as content that scanners should provide, or be able to provide, to applications.

```
{
    "address": [
        {
            "id": "bookName",
            "value": "text"
        },
        {
            "id": "chapterNumber",
            "value": "1 - n"
        },
        {
            "id": "documentNumber",
            "value": "1 - n"
        },
        {
            "id": "imageMerged",
            "value": "yes/no"
        },
        {
            "id": "imageNumber",
            "value": "1 - n"
        },
        {
            "id": "imageSideNumber",
            "value": "1 - n"
        },
        {
            "id": "imageSource",
            "value": "feederFront|feederBack|flatbed"
        },
        {
            "id": "lastSegment",
            "value": "yes/no"
        },
        {
            "id": "patchCode",
            "value": "patch1/patch2/..."
        },
        {
            "id": "segmentNumber",
            "value": "1 - n"
        },
        {
            "id": "sheetNumber",
            "value": "1 - n"
        }
    ],
    "image": [
        {
```

```
            "id": "bitsPerPixel",
            "value": "1 - n"
        },
        {
            "id": "bitsPerSample",
            "value": "1 - n"
        },
        {
            "id": "compression",
            "value": "none/group4/jpeg/..."
        },
        {
            "id": "height",
            "value": "1 - n"
        },
        {
            "id": "pixelFormat",
            "value": "bw1/gray8/rgb24"
        },
        {
            "id": "offsetX",
            "value": "0 - n"
        },
        {
            "id": "offsetY",
            "value": "0 - n"
        },
        {
            "id": "pixelFlavor",
            "value": "whiteonblack/blackonwhite"
        },
        {
            "id": "planar",
            "value": "yes/no"
        },
        {
            "id": "resolution",
            "value": "1 - n"
        },
        {
            "id": "samplesperpixel",
            "value": "1 - n"
        },
        {
            "id": "width",
            "value": "1 - n"
        }
    ]
}
```

Use Cases

This section shows the kinds of user experiences in mind when developing TWAIN Direct.

Home Scanning

John has a cloud aware scanner in his house. Using his favorite tablet he brings up a webpage that connects to his scanner. He has options to save his images to his cloud drive, his local folder, or his favorite social media site. He selects a “taxes” configuration and uses it to save some forms to his local folder.

Mary has an old scanner (one that is not natively TWAIN Direct enabled, but which supports TWAIN and has the TWAIN Direct-to-TWAIN layer). She runs a TWAIN application and uses it to save some preset configurations: one for “photos” and one for old “newspaper” clippings of recipes. Then she brings up her favorite TWAIN Direct scanning application and selects the “photo” option she created and scans with it.

Ad Hoc and Mobile Scanning

Alice goes to the library and uses her phone to access a scanner. Her application is set up by default to request OCR data from the scanner, but to permit scanning to continue if the data cannot be collected. The application captures the images she wants and tells her that there is no text. Alice just wanted the images, so she’s done.

Sam’s office has a shared network scanner. He goes to it and identifies himself. The scanner goes out onto the network and gets his list of configurations. He sees that one is missing and realizes that he’s forgotten to create a new one for himself, so he goes back to his desk and accesses a webpage. The page is served by the scanner vendor and shows all of the features of the scanner. He sets up his configuration and saves it. Then he goes back to the scanner and uses the new configuration to scan his documents.

Susan has a small portable cloud aware scanner that she takes on business trips. She runs an application on her phone. She first selects a configuration to scan receipts. The scanner automatically sends them to her business for processing. She then selects to scan business cards, and these are forwarded to her contacts list.

Small Office Scanning

Jack has a scanner on his desk. He has made arrangements with a cloud vendor to handle his documents. At the start of his day he runs a program that tells the cloud that he is ready to work. For the rest of the day an application in the cloud monitors his device. When it detects that paper is present in the scanner it scans it, analyzes the document and handles it according to rules that he worked out with his vendor.

Service Bureau Scanning

Sally's business scans documents for other companies. She has been using a TWAIN Direct scanner from Acme, but now her scanner is broken and Acme has gone out of business. She finds a TWAIN Direct scanner made by Smith that has the same features. She is able to replace the scanner and does not have to change any of her application code.

Application Development

Bill needs to add scanning to his application. He decides to go with a cloud vendor to discover and register the device. When it comes time to configure the scanner he finds that all he has to do is add a scan command with a request for the best color images, and a destination. He's able to develop and test the complete system in two days, far less than his original expectation.

Carol has an existing application and has been asked to support a new scanner that can detect paperclips. This feature isn't on by default and hasn't been standardized. She decides to go into her existing configurations and add the command to turn it on. Those same configurations can still be used with scanners that don't have the feature, saving her time.

Carl needs a complex configuration, one that collects color and black-and-white images from the scanner, but only from the front, as the back image is black-and-white only and has a barcode that he needs to read. He runs an application provided by the scanner vendor which allows him to create the configuration that he needs. He embeds that configuration (which is nothing more than JSON in text form) in his application and sends it to the device when he wants to scan.

Built on the foundation of the industry-standard TWAIN driver specification, TWAIN Direct is a success-oriented, vendor agnostic solution that enables direct communication between scanning devices and software. From home scanning to Service Bureau production scanning environments, TWAIN Direct will revolutionize how scanners and software communicate.