

1. Description

This document describes the TWAIN 2.0 enhancements and modifications, using TWAIN 1.91 as the basis. Its audience is anyone interested in advancing the TWAIN Specification.

2. History

Date	Reason
06-Dec-2004	First draft.
14-Dec-2004	Added items requested by users
09-Mar-2005	Locked down and cleaned up of incomplete items, from this point on no new additions or changes may be made to this proposal. It is possible for items to be removed.
14-Apr-2005	Added items
28-Apr-2005	Reformatted and clarified some portions of the document

3. Notes

At the time this document was approved by the Technical Subcommittee there were still a number of outstanding TBD items. Most of these TBD items were MICR features. In order to go ahead these items have been removed from the TWAIN 2.0 proposal. This is appropriate because when the subcommittee reviewed the document, approval was given to the completed items. If more items had been resolved, the subcommittee would have review and considered them for approval as well.

TWAIN 2.0 needs to be ratified so that 64-bit support, Linux/Unix and basic MICR support can be added to the standard. All other items – if they result in a delay – will be deferred for consideration in TWAIN 2.1

The previous TWAIN 2.0 proposal will be kept on the website, so that readers can compare this document with that one. Again, items that have been removed from TWAIN 2.0 are only deferred, not permanently out of consideration (though being TBD's they are all in need of further discussion).

At this time this document is stable, with the possible exception of the new additions to ICAP_ORIENTATION, and ICAP_AUTOSIZE, which may be removed if IP analysis of them significantly impacts the ratification schedule.

There is one TBD in this document, and that has to do with the Open Source license we intend to use for the DSM.

4. Overview

TWAIN 2.0 is the planned upgrade from TWAIN 1.91. The decision to increase the major version is based on three factors: the port to Linux/Unix, support for 64-bit operating systems and adoption of Open Source for the DSM (which includes a signification cleanup of the Specification document).

TWAIN 2.0 is backwards compatible to TWAIN 1.5, meaning that applications and drivers that follow the various versions of the specification can interoperate, regardless of what version that is.

The following list summarizes the TWAIN 2.0 modifications.

Feature/Change	Comment
Port to Linux/Unix	Data Source Manager (DSM) to be rewritten and released under appropriate Open Source license. Specification document to be protected by TWAIN Working Group through appropriate licensing and cleaned up so that twain.h file can be derived solely from its content. Twain.h to be cleaned up, removing extraneous comments that belong solely to the TWAIN Specification. Targeted OS's to be determined later, but goal is to permit support for all versions, distributions and variations of Linux/Unix.
64-bit support	DSM to be rewritten to support 64-bit operating systems. Initial target is Windows XP and Windows .NET 2003, but goal is support for all operating systems already supported with a 32-bit DSM. Specification to be modified to describe modifications needed for 64-bit support, goal is to avoid creating new capabilities or operations, if possible. There are no immediate plans for a thinking layer.
PDF/A (ISO 19005) TWFF_PDFA	Support for new ICAP_IMAGEFILEFORMAT – TWFF_PDFA, to be accessed from DAT_SETUPFILEXFER. Specification should direct reader to appropriate ISO reference. Link is included for readers of this document; it will not be included in the TWAIN Specification. <i>http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=38920&scopelist=PROGRAMME</i>
Clarification on relationship between CAP_FEEDERENABLED, ICAP_PHYSICALWIDTH, ICAP_PHYSICALHEIGHT	This is already be covered in the Capability Ordering, but there is a request to clarify that an application changing between the ADF and the Flatbed needs to refresh its understanding of the current physical limits of the cropping area. This should also be extended to ICAP_MINIMUMWIDTH and ICAP_MINIMUMHEIGHT, since it's reasonable for a driver to allow smaller values for the Flatbed than the ADF.
Single Document Multiple Images (SDMI)	Some devices offer the ability to output more than one image from a side of a document. This feature is used to capture images with different characteristics for different kinds of processing in the workflow. For instance, a color document may be captured for archival purposes, while a bitonal document is

****DRAFT****

This document has been locked down for TWAIN 2.0.

	captured for OCR processing, and an infra-red image is captured for defect corrections.
MICR support	Based on a discussion with Harvey Spencer and requests from the TWAIN Developer Community, some MICR related capabilities and operations are being added to the TWAIN Specification to better support check scanning devices and applications.
Add new capabilities.	Added new capabilities and settings to enhance paper handling, DAT_EXTIMAGEINFO handling and simplify identification of custom capability interfaces.
Make key operations and behavior mandatory.	Key capability operations and behavior have been made mandatory in order to improve the dependability of drivers.
Add new DAT_EXTIMAGEINFO items	Extends the defined attributes for DAT_EXTIMAGEINFO to improve the information that can be gleaned from each image on a per page basis.

5. Port to Linux/Unix

5.1. New DSM

The new DSM is a fresh rewrite. The code is Open Source using the [TBD License].

5.2. Specification Cleanup

The TWAIN Specification is protected by the trademark of the TWAIN Working Group. No DSM or twain.h file is able to use the TWAIN name or Logo unless it is compliant to the TWAIN Specification.

Items in twain.h that support the DSM are added to the Specification, to make it possible to recreate the DSM with this document. This information is appropriately marked, since most readers (application and driver writers) are not concerned with it.

To reduce ambiguity, Chapter 8 in the Specification contains a section that defines all the elements needed to construct a twain.h file. This allows anyone to recreate twain.h from the Specification. Constant values in other parts of the document are removed. Most of the comments in the twain.h file are removed.

The embedded application and driver code samples in the Specification are replaced by complete, updated code samples in the appendix section. Text in the Specification refers the reader to the appropriate section in the sample code.

5.3. Testing

The TWAIN Working Group needs testers to confirm the correct behavior of the DSM on any given version or distribution of Linux/Unix.

A section needs to be created on the TWAIN Working Group website that lists the operating systems and versions that TWAIN is confirmed to work on. We have a precedent allowing companies to test internally a beta DSM at their own risk, provided they update to the certified version as soon as it is available.

6. 64-bit Support

6.1. New DSM

The DSM is rewritten for 64-bit versions of Windows XP and Windows .NET 2003.

TWAIN 2.0 brings 64-bit support for all TWAIN supported operating systems (Linux/Unix and Macintosh). Windows and Linux/Unix use the same source code tree (provided under Open Source licensing). Macintosh is supported by Apple.

TWAIN 2.0 does not provide 32-bit/64-bit thunking for any operating system. There is no attempt to describe the thunking interface for legacy 16-bit/32-bit systems in the Specification.

The 64-bit DSM on Windows and Linux requires driver writers to include it in their installations. The full path and file names will be placed in the TWAIN 2.0 specification when it is updated.

6.2. Specification Changes

The TWAIN Specification is modified in places where TW_UINT32 values have been used to hold pointer values (like in DAT_EXTIMAGEINFO). These are replaced with INT_PTR type constructs that allow pointers to be manipulated as integer values, these values are 32-bit (unchanged) when compiled for 32-bit systems and are 64-bit when compile for 64-bit systems.

6.3. Testing

The TWAIN Working Group needs testers to confirm the correct behavior of the 64-bit DSM. We have a precedent allowing companies to test internally a beta DSM at their own risk, provided they update to the certified version as soon as it is available.

7. PDF/A (ISO 19005)

A new image file format TWFF_PDF/A is added to the specification, using the next available number in twain.h. The description in the TWAIN Specification refers the user to ISO/CD 19005-1 for more information about the new format and how to use it.

8. Clarification on relationship between CAP_FEEDERENABLED, ICAP_PHYSICALWIDTH, ICAP_PHYSICALHEIGHT

Clarification has been requested on the relationship between CAP_FEEDERENABLED and ICAP_PHYSICALWIDTH and ICAP_PHYSICALHEIGHT. This request is extended to ICAP_MINIMUMWIDTH and ICAP_MINIMUMHEIGHT. Both the Specification and the Capability Ordering document emphasize that the application should retest these value if it shifts between the ADF and the Flatbed. The clarification follows:

The driver should code defensively, accepting the larger ICAP_PHYSICALWIDTH and ICAP_PHYSICALHEIGHT values if they have not been renegotiated, but internally snapping them to the smaller values. For instance, if an application asks for 12 x 12 for ICAP_FEEDERENABLED set to TRUE, then sets ICAP_FEEDERENABLED to FALSE for a Flatbed that can only handle 8.5 x 12, then the driver should internally snap the width to 8.5. In this example the driver does not have to accept a MSG_SET of 12 for the Flatbed, and it does not have to report a value of 12 for the Flatbed, it simply should work correctly if the application fails to renegotiate the frame values through ICAP_SUPPORTEDSIZES, ICAP_FRAMES or ICAP_SUPPORTEDSIZES.

The driver should handle ICAP_MINIMUMWIDTH and ICAP_MINIMUMHEIGHT in the same fashion if they differ between the ADF and the Flatbed.

9. Single Document Multiple Images (SDMI)

Some devices offer the ability to output more than one image from a side of a document. This feature is used to capture images with different characteristics for different kinds of processing in the workflow. For instance, a color document may be captured for archival purposes, while a bitonal document is captured for OCR processing, and an infra-red image is captured for defect corrections.

This feature depends on what is called camera addressing, the ability to address elements in the device responsible for the various color spaces. To reduce ambiguity the driver must support either `DAT_FILESYSTEM` or `CAP_CAMERASIDE`, and must specifically address the desired camera, even if the device is simplex.

9.1. CAP_CAMERAENABLED

This is a `TW_BOOL` set to `TRUE` or `FALSE`. If set to `TRUE` then the device will deliver images from the specified camera.

9.2. CAP_CAMERAORDER

This is a `TW_ARRAY` containing the `TWPT_(ICAP_PIXELTYPE)` values set in the order they are to be captured by the device. This capability only has meaning if SDMI is in effect. For example, if the device is set to output a `TWPT_RGB` and a `TWPT_BW` from the top (front) side of the document and a `TWPT_RGB` from the rear, and if this capability is set to `{TWPT_BW, TWPT_RGB}`, then the driver will first send the front black-and-white image to the application, then the front color image, then the rear color image.

10. MICR Support

TWAIN has the opportunity to standardize the communication between drivers and applications seeking to do check scanning. The new features focus on both the kind of data passed for check scanning, and the kind of device control needed.

10.1. CAP_MICRENABLED

This capability is a `TW_BOOL` that can be `TRUE` or `FALSE`. Applications should test for its existence to determine if a driver supports check scanning. If enabled then the driver will perform whatever actions are needed to support check scanning.

10.2. CAP_FEEDERPREP

This `TRUE` or `FALSE` feature can be used by any scanner with an ADF. It commands the feeder to perform any action needed to improve the movement of paper through the transport, such as shaking the paper.

10.3. CAP_FEEDERPOCKET

This capability enumerates the available output or collation pockets on the device. A list of numerical TWFP_POCKETS from 1 to 16 are provided, and are organized from time to bottom and left to right, facing in the direction of the motion of the paper. TWFP_POCKETERROR is included too.

10.4. TWEI_MAGTYPE

This describes the kind of magnetic data. For TWAIN 2.0 this is TWMD_MICR only.

10.5. TWEI_MAGDATA

This is a “blob” of data with a byte count retrieved from the driver/device. The interpretation of the data comes from TWEI_MAGTYPE.

10.6. Infra-Red

TWPT_INFRARED is added to ICAP_PIXELTYPE.

10.7. TWCC_Codes

TWAIN has a very small set of error codes that it can deliver during scanning; the proposal is to add more. This discussion occurred during the MICR chat, which is why it is being included here.

Error	Comment
TWCC_INTERLOCK	Cover or door is open
TWCC_DAMAGEDCORNER	Document has a damaged corner
TWCC_FOCUSERROR	Focusing error during document capture
TWCC_DOCTOOLIGHT	Document is too light
TWCC_DOCTOODARK	Document is too dark

11. New Capabilities and Capability Values

11.1. New ICAP_ORIENTATION Values

This section adds new enumerated items, which allow an application to negotiate for automatic rotation of the output image working off the image content to determine the correct Input Orientation. For instance, a document with the letter “A” on it could be captured in any of the four orientations, but will be passed to the application with the image rotated so that “A” will appear in a way the user can naturally read it. TWOR_AUTOTEXT uses text as the criteria for rotation. TWOR_AUTOPICTURE uses a picture as the criteria for rotation. TWOR_AUTO leaves the decision completely up to the device (could be text or picture or anything else).

11.1.1. TWOR_AUTO

The Output image is automatically rotated according to the capabilities of the device.

11.1.2. TWOR_AUTOTEXT

The Output image is automatically rotated to bring text into its proper orientation.

11.1.3. TWOR_AUTOPICTURE

The Output image is automatically rotated to bring a picture into its proper orientation.

11.2. ICAP_AUTOSIZE

This is a new enumeration which allows the application to force image dimensions to match those offered by ICAP_SUPPORTEDSIZES. Note for the application writer that there may also be dependencies between this capability and ICAP_AUTOMATICBORDERDETECTION, ICAP_AUTOMATICDESKEW, ICAP_ORIENTATION, and ICAP_ROTATION.

TWAS_NONE is the default. TWAS_CURRENT snaps to the current ICAP_SUPPORTEDSIZES value. TWAS_AUTO snaps to the nearest match from the ICAP_SUPPORTEDSIZES list.

11.2.1. TWAS_NONE

The capability is disabled (this is the default).

11.2.2. TWAS_CURRENT

Output images are snapped to the current value of ICAP_SUPPORTEDSIZES.

11.2.3. TWAS_AUTO

Output images are snapped to best match from the ICAP_SUPPORTEDSIZES list, this may include custom sizes provided by the driver vendor.

11.3. ICAP_SUPPORTEDEXTIMAGEINFO

Allows an application to know what TWEI_ constants are supported for DAT_EXTIMAGEINFO/MSG_GET calls. The TWAIN Data Source will list all the supported TWEI_ constants in a TW_ARRAY container.

11.4. CAP_CUSTOMINTERFACE GUID

This capability offers applications a significantly more reliable means of uniquely identifying the vendor specific Custom Capability or Operation Interfaces supported by a driver. Currently the only way to uniquely identify a TWAIN Driver is by matching the TW_IDENTITY structure and for many reasons this is unreliable, and there is no reason to assume that TWAIN drivers with a completely different TWAIN identities would not support the same Custom Capability or Operation Interface.

GUID stands for Global Unique Identifier. This capability will return a TW_STR255 in a ONEVALUE container containing a GUID in the form "{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}." This GUID will be used to uniquely identify the Custom Capability or Operation Interface that the driver supports. In particular, the GUID will identify the capabilities that have been defined in the range CAP_CUSTOMBASE whose implementation is determined by the manufacturer.

12. Key Operations and Behavior Made Mandatory

12.1. Supported Values

All Capabilities shall respond with ALL their supported values during the MSG_GET operation, including TW_BOOL type capabilities, etc.

For example, prior to TWAIN 2.0, CAP_FEEDERENABLED / MSG_GET would respond in the identical manner as MSG_GETCURRENT - a TWON_ONEVALUE containing only the current state of the capability.

For TWAIN 2.0 this capability must respond with a TW_ENUMERATION that contains *all* the allowed values, if both TRUE and FALSE are supported then the enumeration would contain a set of values {TRUE, FALSE}.

12.2. DG_CONTROL/DAT_CAPABILITY/ MSG_QUERYSUPPORT

Makes this pre-existing operation mandatory instead of merely recommended or optional. Mandatory requirement of this operation is necessary for testing purposes to determine what features a data source supports. Identifying supported features will be a key point in testing software to determine if a data source adheres to the TWAIN Specification.

13. Add New EXTIMAGEINFO Items

13.1. TWEI_PAGESIDE Constant

Identifies the page side during DAT_EXTIMAGEINFO/MSG_GET calls.

13.2. TWEI_FILESYSTEMSOURCE Constant

This new TWEI_ constant, a TW_STR255, will be used to identify which DAT_FILESYSTEM directory or file name a particular image came from. This is only relevant for data sources that support DAT_FILESYSTEM.