EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

WORKSHOP AGREEMENT

CWA 14050-24

November 2000

ICS 33.160.40; 35.200; 35.240.40

Extensions for Financial Services (XFS) interface specification - Release 3.0 - Part 24: Camera Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

This CEN Workshop Agreement can in no way be held as being an official standard as developed by CEN National Members.

© 2000 CEN All rights of exploitation in any form and by any means reserved world-wide for CEN National Members

Ref. No CWA 14050-24:2000 E

Rue de Stassart, 36 • B-1050 Bruxelles
Tel : +32 2 550 08 11 • Fax : +32 2 550 08 19

Table of Contents

Foreword

This CWA is revision 3.0 of the XFS interface specification.

The move from an XFS 2.0 specification (CWA 13449) to a 3.0 specification has been prompted by a series of factors.

Initially, there has been a technical imperative to extend the scope of the existing specification of the XFS Manager to include new devices, such as the Card Embossing Unit.

Similarly, there has also been pressure, through implementation experience and the advance of the Microsoft technology, to extend the functionality and capabilities of the existing devices covered by the specification.

Finally, it is also clear that our customers and the market are asking for an update to a specification, which is now over 2 years old. Increasing market acceptance and the need to meet this demand is driving the Workshop towards this release.

The clear direction of the CEN/ISSS XFS Workshop, therefore, is the delivery of a new Release 3.0 specification based on a C API. It will be delivered with the promise of the protection of technical investment for existing applications and the design to safeguard future developments.

The CEN/ISSS XFS Workshop gathers suppliers as well as banks and other financial service companies. A list of companies participating in this Workshop and in support of this CWA is available from the CEN/ISSS Secretariat.

This CWA was formally approved by the XFS Workshop meeting on 2000-10-18. The specification is continuously reviewed and commented in the CEN/ISSS Workshop on XFS. It is therefore expected that an update of the specification will be published in due time as a CWA, superseding this revision 3.0.

The CWA is published as a multi-part document, consisting of:

Part 1: Application Programming Interface (API) - Service Provider Interface (SPI); Programmer's Reference

Part 2: Service Classes Definition; Programmer's Reference

Part 3: Printer Device Class Interface - Programmer's Reference

Part 4: Identification Card Device Class Interface - Programmer's Reference

Part 5: Cash Dispenser Device Class Interface - Programmer's Reference

Part 6: PIN Keypad Device Class Interface - Programmer's Reference

Part 7: Check Reader/Scanner Device Class Interface - Programmer's Reference

Part 8: Depository Device Class Interface - Programmer's Reference

Part 9: Text Terminal Unit Device Class Interface - Programmer's Reference

Part 10: Sensors and Indicators Unit Device Class Interface - Programmer's Reference

Part 11: Vendor Dependent Mode Device Class Interface - Programmer's Reference

Part 12: Camera Device Class Interface - Programmer's Reference

Part 13: Alarm Device Class Interface - Programmer's Reference

Part 14: Card Embossing Unit Class Interface - Programmer's Reference

Part 15: Cash In Module Device Class Interface- Programmer's Reference

Part 16: Application Programming Interface (API) - Service Provider Interface (SPI) - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 17: Printer Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 18: Identification Card Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 19: Cash Dispenser Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 20: PIN Keypad Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 21: Depository Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 22: Text Terminal Unit Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 23: Sensors and Indicators Unit Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 24: Camera Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 25: Identification Card Device Class Interface - PC/SC Integration Guidelines

In addition to these Programmer's Reference specifications, the reader of this CWA is also referred to a complementary document, called Release Notes. The Release Notes contain clarifications and explanations on the CWA specifications, which are not requiring functional changes. The current version of the Release Notes is available online from http://www.cenorm.be/isss/Workshop/XFS.

The information in this document represents the Workshop's current views on the issues discussed as of the date of publication. It is furnished for informational purposes only and is subject to change without notice. CEN/ISSS makes no warranty, express or implied, with respect to this document.

1. General

A new reset command, a new media threshold event parameter, individual status values for each camera, and UNICODE support for exposure text data have been added. In addition, the meanings of the various device status values have been clarified.

2. New Chapter

2.1 References

1. XFS Application Programming Interface (API)/Service Provider Interface (SPI), Programmer's Reference Revision 3.0, October 18, 2000

3. New Info Commands

None.

4. Changes to existing Info Commands

4.1 WFS_INF_CAM_STATUS

Description This command reports the full range of information available, including the information that is provided by the service provider.

Input Param None.

Output Param

```
LPWFSCAMSTATUS      lpStatus;

typedef struct _wfs_cam_status
   {
   WORD          fwDevice;
   WORD          fwMedia[WFS_CAM_CAMERAS_SIZE];
   WORD          fwCameras[WFS_CAM_CAMERAS_SIZE];
   USHORT        usPictures[WFS_CAM_CAMERAS_SIZE];
   LPSTR         lpszExtra;
   } WFSCAMSTATUS, * LPWFSCAMSTATUS;
```

fwDevice
Specifies the state of the Camera device as one of the following flags:

Value	Meaning
WFS_CAM_DEVONLINE	The device is online (i.e., powered on and operable).
WFS_CAM_DEVOFFLINE	The device is offline (e.g., the operator has taken the device offline by turning a switch or pulling out the device).
WFS_CAM_DEVPOWEROFF	The device is powered off or physically not connected.
WFS_CAM_DEVNODEVICE	There is no device intended to be there; e.g. this type of self service machine does not contain such a device or it is internally not configured.
WFS_CAM_DEVHWERROR	The device is inoperable due to a hardware error.
WFS_CAM_DEVUSERERROR	The device is inoperable because a person is preventing proper operation.
WFS_CAM_DEVBUSY	The device is busy and not able to process an Execute command at this time.

fwMedia[…]
Specifies the state of the recording media of the cameras. A number of indexes are defined below. The maximum fwMedia index is WFS_CAM_CAMERAS_MAX.

fwMedia[WFS_CAM_ROOM]
Specifies the state of the recording media of the camera that monitors the whole self-service area. Specified as one of the following flags:

Value	Meaning
WFS_CAM_MEDIAOK	The media is in a good state.
WFS_CAM_MEDIAHIGH	The media is almost full (threshold).
WFS_CAM_MEDIAFULL	The media is full.
WFS_CAM_MEDIANOTSUPP	The device does not support sensing the media level.
WFS_CAM_MEDIAUNKNOWN	Due to a hardware error or other condition, the state of the media cannot be determined.

fwMedia[WFS_CAM_PERSON]
Specifies the state of the recording media of the camera that monitors the person standing in front of the self-service machine. Specified as one of the following flags:

Value	Meaning
WFS_CAM_MEDIAOK	The media is in a good state.
WFS_CAM_MEDIAHIGH	The media is almost full (threshold).
WFS_CAM_MEDIAFULL	The media is full.
WFS_CAM_MEDIANOTSUPP	The device does not support sensing the media level.
WFS_CAM_MEDIAUNKNOWN	Due to a hardware error or other condition, the state of the media cannot be determined.

fwMedia[WFS_CAM_EXITSLOT]
Specifies the state of the recording media of the camera that monitors the exit slot(s) of the self-service machine. Specified as one of the following flags:

Value	Meaning
WFS_CAM_MEDIAOK	The media is in a good state.
WFS_CAM_MEDIAHIGH	The media is almost full (threshold).
WFS_CAM_MEDIAFULL	The media is full.
WFS_CAM_MEDIANOTSUPP	The device does not support sensing the media level.
WFS_CAM_MEDIAUNKNOWN	Due to a hardware error or other condition, the state of the media cannot be determined.

fwCameras[...]
Specifies the state of the cameras. A number of cameras are defined below. The maximum camera index is WFS_CAM_CAMERAS_MAX.

fwCameras[WFS_CAM_ROOM]
Specifies the state of the camera that monitors the whole self-service area. Specified as one of the following flags:

Value	Meaning
WFS_CAM_CAMNOTSUPP	The camera is not supported.
WFS_CAM_CAMOK	The camera is in a good state.
WFS_CAM_CAMINOP	The camera is inoperative.
WFS_CAM_CAMUNKNOWN	Due to a hardware error or other condition, the state of the camera cannot be determined.

fwCameras[WFS_CAM_PERSON]
Specifies the state of the camera that monitors the person standing in front of the self-service machine. Specified as one of the following flags:

Value	Meaning
WFS_CAM_CAMNOTSUPP	The camera is not supported.
WFS_CAM_CAMOK	The camera is in a good state.
WFS_CAM_CAMINOP	The camera is inoperative.

WFS_CAM_CAMUNKNOWN	Due to a hardware error or other condition, the state of the camera cannot be determined.

fwCameras[WFS_CAM_EXITSLOT]
Specifies the state of the camera that monitors the exit slot(s) of the self-service machine. Specified as one of the following flags:

Value	Meaning
WFS_CAM_CAMNOTSUPP	The camera is not supported.
WFS_CAM_CAMOK	The camera is in a good state.
WFS_CAM_CAMINOP	The camera is inoperative.
WFS_CAM_CAMUNKNOWN	Due to a hardware error or other condition, the state of the camera cannot be determined.

usPictures[…]
Specifies the number of pictures stored on the recording media of the cameras. A number of indexes are defined below. The maximum *usPictures* index is WFS_CAM_CAMERAS_MAX.

Index	Meaning
WFS_CAM_ROOM	The camera that monitors the whole self-service area.
WFS_CAM_PERSON	The camera that monitors the person standing in front of the self-service machine
WFS_CAM_EXITSLOT	The camera that monitors the exit slot(s) of the self-service machine.

lpszExtra
Specifies a list of vendor-specific, or any other extended, information. The information is returned as a series of "*key=value*" strings so that it is easily extensible by service providers. Each string will be null-terminated, with the final string terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments Applications which require or expect specific information to be present in the *lpszExtra* parameter may not be device or vendor-independent.

4.2 WFS_INF_CAM_CAPABILITIES

Description This command is used to retrieve the capabilities of the Camera System

Input Param None.

Output Param `LPWFSCAMCAPS lpCaps;`

```
typedef struct _wfs_cam_caps
   {
   WORD          wClass;
   WORD          fwType;
   WORD          fwCameras[WFS_CAM_CAMERAS_SIZE];
   USHORT        usMaxPictures;
   WORD          fwCamData;
   USHORT        usMaxDataLength;
   WORD          fwCharSupport;
   LPSTR         lpszExtra;
   } WFSCAMCAPS, * LPWFSCAMCAPS;
```

wClass
Specifies the logical service class, value is:
WFS_SERVICE_CLASS_CAM

fwType
Specifies the type of the camera device; only current value is:

Value	Meaning
WFS_CAM_TYPE_CAM	Camera system

fwCameras[...]
Specifies which cameras are available. A number of cameras are defined below. The maximum camera index is WFS_CAM_CAMERAS_MAX.

fwCameras[WFS_CAM_ROOM]
Specifies whether the camera that monitors the whole self-service area is available. Specified as one of the following flags:

Value	Meaning
WFS_CAM_NOT_AVAILABLE	This camera is not available.
WFS_CAM_AVAILABLE	This camera is available.

fwCameras[WFS_CAM_PERSON]
Specifies whether the camera that monitors the person standing in front of the self-service machine is available. Specified as one of the following flags:

Value	Meaning
WFS_CAM_NOT_AVAILABLE	This camera is not available.
WFS_CAM_AVAILABLE	This camera is available.

fwCameras[WFS_CAM_EXITSLOT]
Specifies whether the camera that monitors the exit slot(s) of the self-service machine is available. Specified as one of the following flags:

Value	Meaning
WFS_CAM_NOT_AVAILABLE	This camera is not available.
WFS_CAM_AVAILABLE	This camera is available.

usMaxPictures
Specifies the maximum number of pictures that can be stored on the recording media.

fwCamData
Specifies, if data can be added to the picture. Specified as a combination of the following flags:

Value	Meaning
WFS_CAM_NOTADD	No data can be added to the picture.
WFS_CAM_AUTOADD	Data is added automatically to the picture.
WFS_CAM_MANADD	Data can be added manually to the picture using the filed *lpszCamData* in the WFS_CMD_CAM_TAKE_PICTURE command.

usMaxDataLength
Specifies the maximum length of the data that is displayed on the photo. Zero, if data cannot be manually added to the picture.

fwCharSupport
One or more flags specifying the Character Set supported by the service provider:

Value	Meaning
WFS_CAM_ASCII	ASCII is supported for execute command data values.
WFS_CAM_UNICODE	UNICODE is supported for execute command data values.

lpszExtra
Specifies a list of vendor-specific, or any other extended, information. The information is returned as a series of "*key=value*" strings so that it is easily extensible by service providers. Each string will be null-terminated, with the final string terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments Applications which require or expect specific information to be present in the *lpszExtra* parameter may not be device or vendor-independent.

5. New Execute Commands

5.1 WFS_CMD_CAM_RESET

Description Sends a service reset to the service provider.

Input Param None

Output Param None.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Events Only the generic events defined in [Ref. 1] can be generated by this command.

Comments This command is used by an application control program to cause a device to reset itself to a known good condition.

6. Changes to existing Execute Commands

6.1 WFS_CMD_CAM_TAKE_PICTURE

Description This command is used to start the recording of the camera system. It is possible to select which camera or which camera position should be used to take a picture. Furthermore data can be sent to be displayed on the photo.

Input Param

```
LPWFSCAMTAKEPICT    lpTakePict;

typedef struct _wfs_cam_take_picture
   {
   WORD          wCamera;
   LPSTR         lpszCamData;
   LPWSTR        lpszUNICODECamData;
   } WFSCAMTAKEPICT, * LPWFSCAMTAKEPICT;
```

wCamera
Specifies the camera that should take the photo as one of the following flags:

Value	Meaning
WFS_CAM_ROOM	Monitors the whole self-service area.
WFS_CAM_PERSON	Monitors the person standing in front of the self-service machine.
WFS_CAM_EXITSLOT	Monitors the exit slot(s) of the self-service machine.

lpszCamData
Specifies the text string to be displayed on the photo. If the maximum text length is exceeded, it will be truncated. In this case or if the text given is invalid an execute event WFS_EXEE_CAM_INVALIDDATA is generated. Nevertheless the picture is taken.

lpszUNICODECamData
Specifies the UNICODE text string to be displayed on the photo. If the maximum text length is exceeded, it will be truncated. In this case or if the text given is invalid an execute event WFS_EXEE_CAM_INVALIDDATA is generated. Nevertheless the picture is taken.

The *lpszUNICODECamData* field should only be used if the service provider supports UNICODE. The *lpszCamData* and *lpszUNICODECamData* fields are mutually exclusive.

Output Param None.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_CAM_CAMNOTSUPP	The specified camera is not supported.
WFS_ERR_CAM_MEDIAFULL	The recording media is full.
WFS_ERR_CAM_CAMINOP	The specified camera is inoperable.

WFS_ERR_CAM_CHARSETNOTSUPP	Character set(s) supported by service provider is inconsistent with use of *lpszCamData* or *lpszUNICODECamData* fields.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_USRE_CAM_MEDIATHRESHOLD	The state of the recording media reached a threshold.
WFS_EXEE_CAM_INVALIDDATA	The text string given is to long or in some other way invalid.

Comments None.

7. New Events

None.

8. Changes to existing Events

8.1 WFS_USRE_CAM_MEDIATHRESHOLD

Description This user event is used to specify that the state of the recording media reached a threshold.

Event Param `LPWORD lpwMediaThreshold;`

Specified as one of the following flags:

Value	Meaning
WFS_CAM_MEDIAOK	The recording media is in a good state.
WFS_CAM_MEDIAHIGH	The recording media is almost full.
WFS_CAM_MEDIAFULL	The recording media is full.

Comments None.

Implementation Advice:
With version 3.0 there is no need to poll the status of the recording media by issuing WFS_INF_CAM_STATUS commands. The application may now receive this event if the Service Provider is able to detect the change.

9. Changes to C-Header file

```
/******************************************************************************
*                                                                             *
* xfscam.h      XFS - Camera (CAM) definitions                                *
*                                                                             *
*               Version 3.00 (10/18/00)                                       *
*                                                                             *
******************************************************************************/

#ifndef __INC_XFSCAM__H
#define __INC_XFSCAM__H

#ifdef __cplusplus
extern "C" {
#endif

#include    <xfsapi.h>

/* be aware of  alignment */
#pragma pack (push, 1)

/* values of WFSCAMCAPS.wClass */

#define     WFS_SERVICE_CLASS_CAM               (10)
#define     WFS_SERVICE_VERSION_CAM             (0x0003) /* Version 3.00 */
#define     WFS_SERVICE_NAME_CAM                "CAM"

#define     CAM_SERVICE_OFFSET                  (WFS_SERVICE_CLASS_CAM * 100)

/* CAM Info Commands */

#define     WFS_INF_CAM_STATUS                  (CAM_SERVICE_OFFSET + 1)
#define     WFS_INF_CAM_CAPABILITIES            (CAM_SERVICE_OFFSET + 2)

/* CAM Execute Commands */

#define     WFS_CMD_CAM_TAKE_PICTURE            (CAM_SERVICE_OFFSET + 1)
#define     WFS_CMD_CAM_RESET                   (CAM_SERVICE_OFFSET + 2)

/* CAM Messages */

#define     WFS_USRE_CAM_MEDIATHRESHOLD         (CAM_SERVICE_OFFSET + 1)
#define     WFS_EXEE_CAM_INVALIDDATA            (CAM_SERVICE_OFFSET + 2)

/* values of WFSCAMSTATUS.fwDevice */

#define     WFS_CAM_DEVONLINE                   WFS_STAT_DEVONLINE
#define     WFS_CAM_DEVOFFLINE                  WFS_STAT_DEVOFFLINE
#define     WFS_CAM_DEVPOWEROFF                 WFS_STAT_DEVPOWEROFF
#define     WFS_CAM_DEVNODEVICE                 WFS_STAT_DEVNODEVICE
#define     WFS_CAM_DEVHWERROR                  WFS_STAT_DEVHWERROR
#define     WFS_CAM_DEVUSERERROR                WFS_STAT_DEVUSERERROR
#define     WFS_CAM_DEVBUSY                     WFS_STAT_DEVBUSY

/* number of cameras supported/length of WFSCAMSTATUS.fwCameras field */

#define     WFS_CAM_CAMERAS_SIZE                (8)
#define     WFS_CAM_CAMERAS_MAX                 (WFS_CAM_CAMERAS_SIZE - 1)

/* indices of WFSCAMSTATUS.fwMedia[...]
              WFSCAMSTATUS.fwCameras [...]
              WFSCAMSTATUS.fwPictures[...]
              WFSCAMCAPS.fwCameras [...]
              WFSCAMTAKEPICT.wCamera             */
#define     WFS_CAM_ROOM                        (0)
#define     WFS_CAM_PERSON                      (1)
#define     WFS_CAM_EXITSLOT                    (2)

/* values of WFSCAMSTATUS.fwMedia */

#define     WFS_CAM_MEDIAOK                     (0)
#define     WFS_CAM_MEDIAHIGH                   (1)
```

```
#define     WFS_CAM_MEDIAFULL                    (2)
#define     WFS_CAM_MEDIAUNKNOWN                 (3)
#define     WFS_CAM_MEDIANOTSUPP                 (4)

/* values of WFSCAMSTATUS.fwCameras */

#define     WFS_CAM_CAMNOTSUPP                   (0)
#define     WFS_CAM_CAMOK                        (1)
#define     WFS_CAM_CAMINOP                      (2)
#define     WFS_CAM_CAMUNKNOWN                   (3)

/* values of WFSCAMCAPS.fwType */

#define     WFS_CAM_TYPE_CAM                     (1)

/* values of WFSCAMCAPS.fwCameras */

#define     WFS_CAM_NOT_AVAILABLE                (0)
#define     WFS_CAM_AVAILABLE                   (1)

/* values of WFSCAMCAPS.fwCamData */

#define     WFS_CAM_NOTADD                       (0)
#define     WFS_CAM_AUTOADD                      (1)
#define     WFS_CAM_MANADD                       (2)

/* values of WFSCAMCAPS.fwCharSupport, WFSCAMTAKEPICT.fwCharSupport  */

#define     WFS_CAM_ASCII                        (0x0001)
#define     WFS_CAM_UNICODE                      (0x0002)


/* XFS CAM Errors */

#define WFS_ERR_CAM_CAMNOTSUPP           (-(CAM_SERVICE_OFFSET + 0))
#define WFS_ERR_CAM_MEDIAFULL            (-(CAM_SERVICE_OFFSET + 1))
#define WFS_ERR_CAM_CAMINOP              (-(CAM_SERVICE_OFFSET + 2))
#define WFS_ERR_CAM_CHARSETNOTSUPP       (-(CAM_SERVICE_OFFSET + 3))


/*=================================================================*/
/* CAM Info Command Structures */
/*=================================================================*/

typedef struct _wfs_cam_status
{
    WORD            fwDevice;
    WORD            fwMedia[WFS_CAM_CAMERAS_SIZE];
    WORD            fwCameras[WFS_CAM_CAMERAS_SIZE];
    USHORT          usPictures[WFS_CAM_CAMERAS_SIZE];
    LPSTR           lpszExtra;
} WFSCAMSTATUS, *LPWFSCAMSTATUS;

typedef struct _wfs_cam_caps
{
    WORD            wClass;
    WORD            fwType;
    WORD            fwCameras[WFS_CAM_CAMERAS_SIZE];
    USHORT          usMaxPictures;
    WORD            fwCamData;
    USHORT          usMaxDataLength;
    WORD            fwCharSupport;
    LPSTR           lpszExtra;
} WFSCAMCAPS, * LPWFSCAMCAPS;


/*=================================================================*/
/* CAM Execute Command Structures */
/*=================================================================*/

typedef struct _wfs_cam_take_picture
{
    WORD            wCamera;
    LPSTR           lpszCamData;
    LPWSTR          lpszUNICODECamData;
} WFSCAMTAKEPICT, *LPWFSCAMTAKEPICT;

/* restore alignment */
#pragma pack (pop)

#ifdef __cplusplus
}       /*extern "C"*/
#endif

#endif  /* __INC_XFSCAM__H */
```