CEN

WORKSHOP

AGREEMENT

CWA 16926-72

February 2020

ICS 35.200; 35.240.15; 35.240.40

English version

Extensions for Financial Services (XFS) interface specification Release 3.40 - Part 72: Alarm Device Class Interface - Migration from version 3.30 (CWA 16926) to Version 3.40 (this CWA) - Programmer's Reference

This CEN Workshop Agreement has been drafted and approved by a Workshop of representatives of interested parties, the constitution of which is indicated in the foreword of this Workshop Agreement.

The formal process followed by the Workshop in the development of this Workshop Agreement has been endorsed by the National Members of CEN but neither the National Members of CEN nor the CEN-CENELEC Management Centre can be held accountable for the technical content of this CEN Workshop Agreement or possible conflicts with standards or legislation.

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

This CEN Workshop Agreement is publicly available as a reference document from the CEN Members National Standard Bodies.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and United Kingdom.

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

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

Ref. No.:CWA 16926-72:2020 E

Table of Contents

European Foreword

This CEN Workshop Agreement has been developed in accordance with the CEN-CENELEC Guide 29 "CEN/CENELEC Workshop Agreements – The way to rapid consensus" and with the relevant provisions of CEN/CENELEC Internal Regulations - Part 2. It was approved by a Workshop of representatives of interested parties on 2019-10-08, the constitution of which was supported by CEN following several public calls for participation, the first of which was made on 1998-06-24. However, this CEN Workshop Agreement does not necessarily include all relevant stakeholders.

The final text of this CEN Workshop Agreement was provided to CEN for publication on 2019-12-12.

The following organizations and individuals developed and approved this CEN Workshop Agreement:

- ATM Japan LTD

- AURIGA SPA

- BANK OF AMERICA

- CASHWAY TECHNOLOGY

- CHINAL ECTRONIC FINANCIAL EQUIPMENT SYSTEM CO.

- CIMA SPA

- CLEAR2PAY SCOTLAND LIMITED

- DIEBOLD NIXDORF

- EASTERN COMMUNICATIONS CO. LTD – EASTCOM

- FINANZ INFORMATIK

- FUJITSU FRONTECH LIMITED

- FUJITSU TECHNOLOGY

- GLORY LTD

- GRG BANKING EQUIPMENT HK CO LTD

- HESS CASH SYSTEMS GMBH & CO. KG

- HITACHI OMRON TS CORP.

- HYOSUNG TNS INC

- JIANGSU GUOGUANG ELECTRONIC INFORMATION TECHNOLOGY

- KAL

- KEBA AG

- NCR FSG

- NEC CORPORATION

- OKI ELECTRIC INDUSTRY SHENZHEN

- OKI ELECTRONIC INDUSTRY CO

- PERTO S/A

- REINER GMBH & CO KG

- SALZBURGER BANKEN SOFTWARE

- SIGMA SPA

- TEB

- ZIJIN FULCRUM TECHNOLOGY CO

It is possible that some elements of this CEN/CWA may be subject to patent rights. The CEN-CENELEC policy on patent rights is set out in CEN-CENELEC Guide 8 "Guidelines for Implementation of the Common IPR Policy on Patents (and other statutory intellectual property rights based on inventions)". CEN shall not be held responsible for identifying any or all such patent rights.

The Workshop participants have made every effort to ensure the reliability and accuracy of the technical and non-technical content of CWA 16926-72, but this does not guarantee, either explicitly or implicitly, its correctness. Users of CWA 16926-72should be aware that neither the Workshop participants, nor CEN can be held liable for damages or losses of any kind whatsoever which may arise from its application. Users of CWA 16926-72do so on their own responsibility and at their own risk.

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 and Scanning 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 Device Class Interface - Programmer's Reference

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

Part 16: Card Dispenser Device Class Interface - Programmer's Reference

Part 17: Barcode Reader Device Class Interface - Programmer's Reference

Part 18: Item Processing Module Device Class Interface - Programmer's Reference

Part 19: Biometrics Device Class Interface - Programmer's Reference

Parts 20 - 28: Reserved for future use.

Parts 29 through 47 constitute an optional addendum to this CWA. They define the integration between the SNMP standard and the set of status and statistical information exported by the Service Providers.

Part 29: XFS MIB Architecture and SNMP Extensions - Programmer's Reference

Part 30: XFS MIB Device Specific Definitions - Printer Device Class

Part 31: XFS MIB Device Specific Definitions - Identification Card Device Class

Part 32: XFS MIB Device Specific Definitions - Cash Dispenser Device Class

Part 33: XFS MIB Device Specific Definitions - PIN Keypad Device Class

Part 76: Barcode Reader Device Class Interface - Migration from Version 3.30 (CWA 16926) to Version 3.40 (this CWA) - Programmer's Reference

Part 77: Item Processing Module Device Class Interface - Migration from Version 3.30 (CWA 16926) to Version 3.40 (this CWA) - Programmer's Reference

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: https://www.cen.eu/work/Sectors/Digital_society/Pages/WSXFS.aspx.

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

1. Migration Information

XFS 3.40 has been designed to minimize backwards compatibility issues. This document highlights the changes made to the Alarm (ALM) Specification device class between version 3.30 and 3.40, by highlighting the additions and deletions to the text.

2. Alarms

This specification describes the functionality of the services provided by Alarms (ALM) under XFS, by defining the service-specific commands that can be issued, using the **WFSGetInfo, WFSAsyncGetInfo**, **WFSExecute** and **WFSAsyncExecute** functions. This section describes the functionality of an Alarm (ALM) service that applies to both attended and unattended (self-service) devices.

The Alarm device class is provided as a separate service due to the need to set or reset an Alarm when one or more logical services associated with an attended CDM or unattended (self-service) device are locked. Because logical services can be locked by the application the Alarm is implemented in a separate device class to ensure that a set (trigger) or reset operation can be performed at any time.

3. References

1. XFS Application Programming Interface (API)/Service Provider Interface (SPI), Programmer's Reference Revision 3.40

4. Info Commands

4.1 WFS_INF_ALM_STATUS

Description   This command is used to request the Alarm status.

Input Param   None.

Output Param   LPWFSALMSTATUS lpStatus;

```
typedef struct _wfs_alm_status
    {
    WORD                fwDevice;
    BOOL                bAlarmSet;
    LPSTR               lpszExtra;
    WORD                wAntiFraudModule;
    } WFSALMSTATUS, *LPWFSALMSTATUS;
```

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

Value	Meaning
WFS_ALM_DEVONLINE	The device is present, powered on and online (i.e. operational, not busy processing a request and not in an error state).
WFS_ALM_DEVOFFLINE	The device is offline (e.g. the operator has taken the device offline by turning a switch).
WFS_ALM_DEVPOWEROFF	The device is powered off or physically not connected.
WFS_ALM_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_ALM_DEVUSERERROR	The device is present but a person is preventing proper device operation. The application should suspend the device operation or remove the device from service until the Service Provider generates a device state change event indicating the condition of the device has changed e.g. the error is removed (WFS_ALM_DEVONLINE) or a permanent error condition has occurred (WFS_ALM_DEVHWERROR).
WFS_ALM_DEVHWERROR	The device is present but inoperable due to a hardware fault that prevents it from being used.
WFS_ALM_DEVBUSY	The device is busy and unable to process an execute command at this time.
WFS_ALM_DEVFRAUDATTEMPT	The device is present but is inoperable because it has detected a fraud attempt.
WFS_ALM_DEVPOTENTIALFRAUD	The device has detected a potential fraud attempt and is capable of remaining in service. In this case the application should make the decision as to whether to take the device offline.

bAlarmSet
Specifies the state of the Alarm as either Reset (FALSE) or Set (TRUE).

lpszExtra
Pointer to 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 is null-terminated, with the final string terminating with two null characters. An empty list may be indicated by either a NULL pointer or a pointer to two consecutive null characters.

wAntiFraudModule
Specifies the state of the anti-fraud module as one of the following values:

Value	Meaning
WFS_ALM_AFMNOTSUPP	No anti-fraud module is available.
WFS_ALM_AFMOK	Anti-fraud module is in a good state and no foreign device is detected.
WFS_ALM_AFMINOP	Anti-fraud module is inoperable.
WFS_ALM_AFMDEVICEDETECTED	Anti-fraud module detected the presence of a foreign device.
WFS_ALM_AFMUNKNOWN	The state of the anti-fraud module cannot be determined.

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.

In the case where communications with the device has been lost, the *fwDevice* field will report WFS_ALM_DEVPOWEROFF when the device has been removed or WFS_ALM_DEVHWERROR if the communications are unexpectedly lost. All other fields should contain a value based on the following rules and priority:

1. Report the value as unknown.

2. Report the value as a general h/w error.

3. Report the value as the last known value.

4.2   WFS_INF_ALM_CAPABILITIES

Description   This command is used to retrieve the capabilities of the Alarm.

Input Param   None.

Output Param   LPWFSALMCAPS lpCaps;

```
typedef struct _wfs_alm_caps
    {
    WORD                wClass;
    BOOL                bProgrammaticallyDeactivate;
    LPSTR               lpszExtra;
    BOOL                bAntiFraudModule;
    LPDWORD             lpdwSynchronizableCommands;
    } WFSALMCAPS, *LPWFSALMCAPS;
```

wClass
Specifies the logical service class as WFS_SERVICE_CLASS_ALM.

bProgrammaticallyDeactivate
Specifies whether the Alarm can be programmatically deactivated (TRUE) or can not be
programmatically deactivated (FALSE).

lpszExtra
Pointer to 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 is null-terminated, with the final string terminating with two null characters. An
empty list may be indicated by either a NULL pointer or a pointer to two consecutive null
characters.

bAntiFraudModule
Specifies whether the anti-fraud module is available. This can either be TRUE if available or
FALSE if not available.

lpdwSynchronizableCommands
Pointer to a zero-terminated list of DWORDs which contains the execute command IDs that can
be synchronized. If no execute command can be synchronized then this parameter will be NULL.

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. Execute Commands

5.1   WFS_CMD_ALM_SET_ALARM

Description   This command is used to trigger an Alarm.

Input Param   None.

Output Param   None.

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

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

Value	Meaning
WFS_SRVE_ALM_DEVICE_SET	The alarm device has been triggered.

Comments   None.

5.2   WFS_CMD_ALM_RESET_ALARM

Description   This command is used to reset an Alarm.

Input Param   None.

Output Param   None.

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

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

Value	Meaning
WFS_SRVE_ALM_DEVICE_RESET	The alarm device has been reset.

Comments   None.

5.3   WFS_CMD_ALM_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.

5.4   WFS_CMD_ALM_SYNCHRONIZE_COMMAND

Description   This command is used to reduce response time of a command (e.g. for synchronization with display) as well as to synchronize actions of the different device classes. This command is intended to be used only on hardware which is capable of synchronizing functionality within a single device class or with other device classes.

The list of execute commands which this command supports for synchronization is retrieved in the *lpdwSynchronizableCommands* parameter of the WFS_INF_ALM_CAPABILITIES.

This command is optional, i.e. any other command can be called without having to call it in advance. Any preparation that occurs by calling this command will not affect any other subsequent command. However, any subsequent execute command other than the one that was specified in the *dwCommand* input parameter will execute normally and may invalidate the pending synchronization. In this case the application should call the WFS_CMD_ALM_SYNCHRONIZE_COMMAND again in order to start a synchronization.

Input Param   LPWFSALMSYNCHRONIZECOMMAND lpSynchronizeCommand;

```
typedef struct _wfs_alm_synchronize_command
    {
    DWORD                   dwCommand;
    LPVOID                  lpCmdData;
    } WFSALMSYNCHRONIZECOMMAND, *LPWFSALMSYNCHRONIZECOMMAND;
```

dwCommand
The command ID of the command to be synchronized and executed next.

lpCmdData
Pointer to data or a data structure that represents the parameter that is normally associated with the command that is specified in *dwCommand*. This parameter can be NULL if no command input parameter is needed or if this detail is not needed to synchronize for the command.

It will be device-dependent whether the synchronization is effective or not in the case where the application synchronizes for a command with this command specifying a parameter but subsequently executes the synchronized command with a different parameter. This case should not result in an error; however, the preparation effect could be different from what the application expects. The application should, therefore, make sure to use the same parameter between *lpCmdData* of this command and the subsequent corresponding execute command.

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_ALM_COMMANDUNSUPP	The command specified in the *dwCommand* field is not supported by the Service Provider.
WFS_ERR_ALM_SYNCHRONIZEUNSUPP	The preparation for the command specified in the *dwCommand* with the parameter specified in the *lpCmdData* is not supported by the Service Provider.

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

Comments   For sample flows of this synchronization see the [Ref 1] Appendix C.

6. Events

6.1 WFS_SRVE_ALM_DEVICE_SET

Description The Alarm has been set (triggered) by an external event or a programmatic request to set (trigger) the Alarm.

Event Param None.

Comments None.

6.2   WFS_SRVE_ALM_DEVICE_RESET

Description   The Alarm has been manually or programmatically reset.

Event Param   None.

Comments   None.

7. C - Header file

```
/****************************************************************************
*                                                                          *
* xfsalm.h      XFS – Alarm (ALM) definitions                              *
*                                                                          *
*                Version 3.40   (December 6 2019)                          *
*                                                                          *
****************************************************************************/

#ifndef __INC_XFSALM__H
#define __INC_XFSALM__H

#ifdef __cplusplus
extern "C" {
#endif

#include <xfsapi.h>

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

/* values of WFSALMCAPS.wClass */

#define     WFS_SERVICE_CLASS_ALM              (11)
#define     WFS_SERVICE_CLASS_VERSION_ALM      (0x2803) /*Version 3.40 */
#define     WFS_SERVICE_CLASS_NAME_ALM         "ALM"

#define     ALM_SERVICE_OFFSET                 (WFS_SERVICE_CLASS_ALM * 100)

/* ALM Info Commands */

#define     WFS_INF_ALM_STATUS                 (ALM_SERVICE_OFFSET + 1)
#define     WFS_INF_ALM_CAPABILITIES           (ALM_SERVICE_OFFSET + 2)

/* ALM Execute Commands */

#define     WFS_CMD_ALM_SET_ALARM              (ALM_SERVICE_OFFSET + 1)
#define     WFS_CMD_ALM_RESET_ALARM            (ALM_SERVICE_OFFSET + 2)
#define     WFS_CMD_ALM_RESET                  (ALM_SERVICE_OFFSET + 3)
#define     WFS_CMD_ALM_SYNCHRONIZE_COMMAND    (ALM_SERVICE_OFFSET + 4)

/* ALM Messages */

#define     WFS_SRVE_ALM_DEVICE_SET            (ALM_SERVICE_OFFSET + 1)
#define     WFS_SRVE_ALM_DEVICE_RESET          (ALM_SERVICE_OFFSET + 2)

/* values of WFSALMSTATUS.fwDevice */

#define     WFS_ALM_DEVONLINE                  WFS_STAT_DEVONLINE
#define     WFS_ALM_DEVOFFLINE                 WFS_STAT_DEVOFFLINE
#define     WFS_ALM_DEVPOWEROFF                WFS_STAT_DEVPOWEROFF
#define     WFS_ALM_DEVNODEVICE                WFS_STAT_DEVNODEVICE
#define     WFS_ALM_DEVHWERROR                 WFS_STAT_DEVHWERROR
#define     WFS_ALM_DEVUSERERROR               WFS_STAT_DEVUSERERROR
#define     WFS_ALM_DEVBUSY                    WFS_STAT_DEVBUSY
#define     WFS_ALM_DEVFRAUDATTEMPT            WFS_STAT_DEVFRAUDATTEMPT
#define     WFS_ALM_DEVPOTENTIALFRAUD          WFS_STAT_DEVPOTENTIALFRAUD

/* values of WFSALMSTATUS.wAntiFraudModule */

#define     WFS_ALM_AFMNOTSUPP                 (0)
#define     WFS_ALM_AFMOK                      (1)
#define     WFS_ALM_AFMINOP                    (2)
#define     WFS_ALM_AFMDEVICEDETECTED          (3)
#define     WFS_ALM_AFMUNKNOWN                 (4)

/* XFS ALM Errors */
```

```
#define WFS_ERR_ALM_COMMANDUNSUPP                (-(ALM_SERVICE_OFFSET + 0))
#define WFS_ERR_ALM_SYNCHRONIZEUNSUPP            (-(ALM_SERVICE_OFFSET + 1))


/*==================================================================*/
/* ALM Info Command Structures */
/*==================================================================*/

typedef struct _wfs_alm_status
{
    WORD                fwDevice;
    BOOL                bAlarmSet;
    LPSTR               lpszExtra;
    WORD                wAntiFraudModule;
} WFSALMSTATUS, *LPWFSALMSTATUS;

typedef struct _wfs_alm_caps
{
    WORD                wClass;
    BOOL                bProgrammaticallyDeactivate;
    LPSTR               lpszExtra;
    BOOL                bAntiFraudModule;
    LPDWORD             lpdwSynchronizableCommands;
} WFSALMCAPS, *LPWFSALMCAPS;

typedef struct _wfs_alm_synchronize_command
{
    DWORD               dwCommand;
    LPVOID              lpCmdData;
} WFSALMSYNCHRONIZECOMMAND, *LPWFSALMSYNCHRONIZECOMMAND;

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

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