CEN

WORKSHOP

AGREEMENT

CWA 16926-2

August 2015

ICS 35.200; 35.240.15; 35.240.40

English version

Extensions for Financial Services (XFS) interface specification Release 3.30 - Part 2: Service Class Definition - 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, Former Yugoslav Republic of Macedonia, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Romania, 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: Avenue Marnix 17, B-1000 Brussels

Table of Contents

European foreword

This CWA is revision 3.30 of the XFS interface specification.

This CEN Workshop Agreement has been drafted and approved by a Workshop of representatives of interested parties on March 19th 2015, the constitution of which was supported by CEN following the public call for participation made on 1998-06-24. 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 3.30.

A list of the individuals and organizations which supported the technical consensus represented by the CEN Workshop Agreement is available from the CEN/XFS Secretariat. The CEN XFS Workshop gathered suppliers as well as banks and other financial service companies.

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

Parts 19 - 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 34: XFS MIB Device Specific Definitions - Check Reader/Scanner Device Class

Part 35: XFS MIB Device Specific Definitions - Depository Device Class

Part 36: XFS MIB Device Specific Definitions - Text Terminal Unit Device Class

Part 37: XFS MIB Device Specific Definitions - Sensors and Indicators Unit Device Class

Part 38: XFS MIB Device Specific Definitions - Camera Device Class

Part 39: XFS MIB Device Specific Definitions - Alarm Device Class

Part 40: XFS MIB Device Specific Definitions - Card Embossing Unit Class

Part 41: XFS MIB Device Specific Definitions - Cash-In Module Device Class

Part 42: Reserved for future use.

Part 43: XFS MIB Device Specific Definitions - Vendor Dependent Mode Device Class

Part 44: XFS MIB Application Management

Part 45: XFS MIB Device Specific Definitions - Card Dispenser Device Class

Part 46: XFS MIB Device Specific Definitions - Barcode Reader Device Class

Part 47: XFS MIB Device Specific Definitions - Item Processing Module Device Class

Parts 48 - 60 are reserved for future use.

Part 61: Application Programming Interface (API) - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Service Provider Interface (SPI) - Programmer's Reference

Part 62: Printer and Scanning Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 63: Identification Card Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 64: Cash Dispenser Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 65: PIN Keypad Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 66: Check Reader/Scanner Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 67: Depository Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 68: Text Terminal Unit Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 69: Sensors and Indicators Unit Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 70: Vendor Dependent Mode Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 71: Camera Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 72: Alarm Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 73: Card Embossing Unit Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 74: Cash-In Module Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 75: Card Dispenser Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 76: Barcode Reader Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (this CWA) - Programmer's Reference

Part 77: Item Processing Module Device Class Interface - Migration from Version 3.20 (CWA 16374) to Version 3.30 (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 http://www.cen.eu/work/areas/ict/ebusiness/pages/ws-xfs.aspx.

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 makes no warranty, express or implied, with respect to this document.

The formal process followed by the Workshop in the development of the CEN 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 the CEN Workshop Agreement or possible conflict 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.

The final review/endorsement round for this CWA was started on 2015-01-16 and was successfully closed on 2015-03-19. The final text of this CWA was submitted to CEN for publication on 2015-06-19. The specification is continuously reviewed and commented in the CEN 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 3.30.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN [and/or CENELEC] shall not be held responsible for identifying any or all such patent rights.

According to the CEN-CENELEC Internal Regulations, the national standards organizations of the following countries are bound to implement this European Standard: Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Former Yugoslav Republic of Macedonia, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and the United Kingdom.

Comments or suggestions from the users of the CEN Workshop Agreement are welcome and should be addressed to the CEN-CENELEC Management Centre.

Revision History:

2.0	November 11, 1996	Initial release of Service Class Document.
3.0	October 18, 2000	Addition of: Alarms (ALM) Card Embossing Unit (CEU) Cash-In Module (CIM) Check Readers and Scanners (CHK)
3.10	November 29, 2007	Addition of: Card Dispenser (CRD) Barcode Reader (BCR) Item Processing Module (IPM)
3.20	March 2, 2011	Release of 3.20. No new device classes.
3.30	March 19, 2015	Release of 3.30. No new device classes.

1. Introduction

1.1   Background to Release 3.30

The CEN/XFS Workshop aims to promote a clear and unambiguous specification defining a multi-vendor software interface to financial peripheral devices. The XFS (eXtensions for Financial Services) specifications are developed within the CEN (European Committee for Standardization/Information Society Standardization System) Workshop environment. CEN Workshops aim to arrive at a European consensus on an issue that can be published as a CEN Workshop Agreement (CWA).

The CEN/XFS Workshop encourages the participation of both banks and vendors in the deliberations required to create an industry standard. The CEN/XFS Workshop achieves its goals by focused sub-groups working electronically and meeting quarterly.

Release 3.30 of the XFS specification is based on a C API and is delivered with the continued promise for the protection of technical investment for existing applications. This release of the specification extends the functionality and capabilities of the existing devices covered by the specification, but it does not include any new device classes. Notable enhancements include:

* Enhanced reporting of Shutter Jammed Status and a new Shutter Status event for CDM, CIM and IPM.
* Addition of a Synchronize command for all device classes, in order to allow synchronized action where necessary.
* Directional Guidance Light support.
* Addition of a CIM Deplete Command.
* Support for EMV Intelligent Contactless Readers.
* Support in PIN for Encrypting Touch Screen.
* PIN Authentication functionality.
* New PIN Encryption Protocols added for Chinese market.
* PIN TR34 standard supported.

1.2   XFS Service-Specific Programming

The service classes are defined by their service-specific commands and the associated data structures, error codes, messages, etc. These commands are used to request functions that are specific to one or more classes of Service Providers, but not all of them, and therefore are not included in the common API for basic or administration functions.

When a service-specific command is common among two or more classes of Service Providers, the syntax of the command is as similar as possible across all services, since a major objective of XFS is to standardize function codes and structures for the broadest variety of services. For example, using the **WFSExecute** function, the commands to read data from various services are as similar as possible to each other in their syntax and data structures.

In general, the specific command set for a service class is defined as a superset of the specific capabilities likely to be provided by the developers of the services of that class; thus any particular device will normally support only a subset of the defined command set.

There are three cases in which a Service Provider may receive a service-specific command that it does not support:

The requested capability is defined for the class of Service Providers by the XFS specification, the particular vendor implementation of that service does not support it, and the unsupported capability is *not* considered to be fundamental to the service. In this case, the Service Provider returns a successful completion, but does no operation. An example would be a request from an application to turn on a control indicator on a passbook printer; the Service Provider recognizes the command, but since the passbook printer it is managing does not include that indicator, the Service Provider does no operation and returns a successful completion to the application.

The requested capability is defined for the class of Service Providers by the XFS specification, the particular vendor

implementation of that service does not support it, and the unsupported capability *is* considered to be fundamental to the service. In this case, a WFS_ERR_UNSUPP_COMMAND error is returned to the calling application. An example would be a request from an application to a cash dispenser to dispense coins; the Service Provider recognizes the command but, since the cash dispenser it is managing dispenses only notes, returns this error.

The requested capability is *not* defined for the class of Service Providers by the XFS specification. In this case, a WFS_ERR_INVALID_COMMAND error is returned to the calling application.

This design allows implementation of applications that can be used with a range of services that provide differing subsets of the functionalities that are defined for their service class. Applications may use the **WFSGetInfo** and **WFSAsyncGetInfo** commands to inquire about the capabilities of the service they are about to use, and modify their behavior accordingly, or they may use functions and then deal with WFS_ERR_UNSUPP_COMMAND error returns to make decisions as to how to use the service.

2. Service Classes

The data and methods needed for the support of self-service, unattended, operations have been defined for XFS (eXtensions for Financial Services) within the following device classes:

- Printer and Scanners

- Identification Card Units

- Cash Dispensers

- Personal Identification Number Keypads (PIN pads)

- Depository Units

- Text Terminal Units

- Sensors and Indicators Units

- Vendor Dependent Mode

- Cameras

- Card Embossing Units

- Alarms

- Cash-In Modules

- Card Dispensers

- Barcode Readers

- Item Processing Modules

The following sections detail for each of the service classes defined for this version of CEN/XFS:

- the standard values to be used as *class* attribute in the configuration information

- the unique number assigned to each service class

- the types of devices defined and supported by the service class specifications

The table below summarizes the unique attributes of each service class:

Service Class	Class Name	Class Identifier	Reference
Printers	PTR	1	CWA XXXXX- 3
Identification Card Units	IDC	2	CWA XXXXX- 4
Cash Dispensers	CDM	3	CWA XXXXX- 5
PIN pads	PIN	4	CWA XXXXX- 6
Check Readers and Scanners	CHK	5	CWA XXXXX- 7
Depository Units	DEP	6	CWA XXXXX- 8
Text Terminal Units	TTU	7	CWA XXXXX- 9
Sensors and Indicators Units	SIU	8	CWA XXXXX- 10
Vendor Dependent Mode	VDM	9	CWA XXXXX- 11
Cameras	CAM	10	CWA XXXXX- 12
Alarms	ALM	11	CWA XXXXX- 13
Card Embossing Units	CEU	12	CWA XXXXX- 14
Cash-In Modules	CIM	13	CWA XXXXX- 15
Card Dispensers	CRD	14	CWA XXXXX- 16
Barcode Readers	BCR	15	CWA XXXXX- 17
Item Processing Modules	IPM	16	CWA XXXXX- 18

2.1   Printers and Scanners (PTR)

Class Name   **PTR**

Class Identifier   WFS_SERVICE_CLASS_PTR = 1

This specification describes the functions provided by a Printer and Scanning (PTR) service.

The XFS printer service defines and supports five types of banking printers through a common interface:

- **Receipt Printer**

  The receipt printer is used to print cut sheet documents. It may or may not require insert or eject operations, and often includes an operator identification device, e.g. Teller A and Teller B lights, for shared operation.

- **Journal Printer**

  The journal is a continuous form device used to record a hardcopy audit trail of transactions, and for certain report printing requirements.

- **Passbook Printer**

  The passbook device is physically and functionally the most complex printer. The XFS definition supports automatic positioning of the book, as well as read/write capability for an optional integrated magnetic stripe. The implementation also manages the book geometry - i.e. the margins and centerfolds - presenting the simplest possible application interface while delivering the full range of functionality.

  Some passbook devices also support the dispensing of new passbooks from up to four passbook paper sources (upper, aux, aux2, lower). Some passbook devices may also be able to place a full passbook in a parking station, print the new passbook and return both to the customer. Passbooks can only be dispensed or moved from the parking station if there is no other media in the print position or in the entry/exit slot.

- **Document Printer**

  Document printing is similar to receipt printing - a set of fields are positioned on one or more inserted sheets of paper - but the focus is on full-size forms. It should be noted that the XFS environment supports the printing of text and graphic fields from the application. The electronic printing of the form image (the template portion of the form which is usually pre-printed with dot-matrix style printers) may also be printed by the application.

- **Scanner Printer**

  The scanner printer is a device incorporating both the capabilities to scan inserted documents and optionally to print on them. These devices may have more than one area where documents may be retained.

Additional hardware components, like scanners, stripe readers, OCR readers, and stamps, normally attached directly to the printer are also controlled through this interface. Additionally the Printer and Scanning class interface can also be used for devices that are capable of scanning without necessarily printing. This includes devices such as Check Scanners.

The specification refers to the terms paper and media. When the term paper is used this refers to paper that is situated in a paper supply attached to the device. The term media is used for media that is inserted by the customer (e.g. check and other material that is scanned) or that is issued to the customer (e.g. a receipt or statement). Receipt, document printers and also passbook printers with white passbook dispensing capability have both. As soon as the paper gets printed it becomes media. Scanners only have media. The term media does not apply to journal printers. When paper is in the print position it is classified as media, on some printers that maintain paper under the print head there will always be both media and paper.

2.2   Identification Card Units (IDC)

Class Name   **IDC**

Class Identifier   **WFS_SERVICE_CLASS_IDC = 2**

This specification describes the functions provided by an Identification Card (IDC) service.

This service allows for the operation of the following categories of units:

- motor driven card reader/writer

- pull through card reader (writing facilities only partially included)

- dip reader

- contactless chip card readers

- permanent chip card readers (each chip is accessed through a unique logical service)

The following tracks/chips and the corresponding international standards are taken into account in this document:

- Track 1   ISO 7811

- Track 2   ISO 7811

- Track 3   ISO 7811 / ISO 4909

- Front Track 1   (JIS II) Japan

- Watermark   Sweden

- Chip (contacted)   ISO 7816

- Chip (contactless)   ISO 10536.

National standards like Transac for France are not considered, but can be easily included via the forms mechanism (see Section 7, Form Definition).

In addition to the pure reading of the tracks mentioned above, security boxes can be used via this service to check the data of writable tracks for manipulation. These boxes (such as CIM or MM) are sensor-equipped devices that are able to check some other information on the card and compare it with the track data.

Persistent values are maintained through power failures, open sessions, close session and system resets.

When the service controls a permanently connected chip card, WFS_ERR_UNSUPP_COMMAND will be returned to all commands except WFS_INF_IDC_STATUS, WFS_INF_IDC_CAPABILITIES, WFS_CMD_IDC_CHIP_POWER, WFS_CMD_IDC_CHP_IO and WFS_CMD_IDC_RESET.

The following defines the roles and responsibilities of an application within EMV:

- EMV Level 2 interaction is handled above the XFS API

- EMV Level 1 interaction is handled below the XFS API

All EMV status information that is defined as a Level 1 responsibility in the EMV specification should be handled below the XFS API.

2.3   Cash Dispensers (CDM)

Class Name   **CDM**

Class Identifier   WFS_SERVICE_CLASS_CDM = 3

This specification describes the functions provided by a Cash Dispenser Module (CDM) service.

Persistent values are maintained through power failures, open sessions, close session and system resets.

This specification covers the dispensing of items. An "item" is defined as any media that can be dispensed and includes coupons, documents, bills and coins. However, if coins and bills are both to be dispensed separate Service Providers must be implemented for each.

All currency parameters in this specification are expressed as a quantity of minimum dispense units, as defined in the description of the WFS_INF_CDM_CURRENCY_EXP command.

There are two types of CDM: Self-Service CDM and Teller CDM. A Self-Service CDM operates in an automated environment, while a Teller CDM has an operator present. The functionality provided by the following commands is only applicable to a Teller CDM:

WFS_CMD_CDM_SET_TELLER_INFO
WFS_INF_CDM_TELLER_INFO

It is possible for the CDM to be part of a compound device with the Cash-In Module (CIM). This CIM\CDM combination is referred to throughout this specification as a "Cash Recycler".

If the device is a Cash Recycler then, if cash unit exchanges are required on both interfaces, the exchanges cannot be performed concurrently. An exchange on one interface must be complete (the WFS_CMD_CDM_END_EXCHANGE must have completed) before an exchange can start on the other interface. The WFS_ERR_CDM_EXCHANGEACTIVE error code will be returned if the correct sequence is not adhered to.

The CIM interface can be used for all exchange operations on recycle devices, and the CIM interface should be used if the device has recycle units of multiple currencies and/or denominations (including multiple note identifiers associated with the same denomination).

The event WFS_SRVE_CDM_COUNTS_CHANGED will be posted if an operation on the CIM interface affects the cash unit counts which are available through the CDM interface.

The following commands on the CIM interface may affect the CDM counts:

> WFS_CMD_CIM_CASH_IN
> WFS_CMD_CIM_CASH_IN_ROLLBACK
> WFS_CMD_CIM_RETRACT
> WFS_CMD_CIM_SET_CASH_IN_UNIT_INFO
> WFS_CMD_CIM_END_EXCHANGE
> WFS_CMD_CIM_RESET

2.4 Personal Identification Number Keypads (PIN)

Class Name   PIN

Class Identifier WFS_SERVICE_CLASS_PIN = 4

This specification describes the functions provided by a Personal Identification Number keypad (PIN) service.

This section describes the general interface for the following functions:

- Administration of encryption devices
- Loading of encryption keys
- Encryption / decryption
- Entering Personal Identification Numbers (PINs)
- PIN verification
- PIN block generation (encrypted PIN)
- Clear text data handling
- Function key handling
- PIN presentation to chipcard
- Read and write safety critical Terminal Data from/to HSM
- HSM and Chipcard Authentication
- EMV 4.0 PIN blocks, EMV 4.0 public key loading, static and dynamic data verification

If the PIN Pad device has local display capability, display handling should be handled using the Text Terminal Unit (TTU) interface.

The adoption of this specification does not imply the adoption of a specific security standard.

Important Notes:

- This revision of this specification does not define all key management procedures; some key management is still vendor-specific.
- Key space management is customer-specific, and is therefore handled by vendor-specific mechanisms.
- Only numeric PIN pads are handled in this specification.

This specification also supports the Hardware Security Module (HSM), which is necessary for the German ZKA Electronic Purse transactions. Furthermore the HSM stores terminal specific data.

This data will be compared against the message data fields (Sent and Received ISO8583 messages) prior to HSM-MAC generation/verification. HSM-MACs are generated/verified only if the message fields match the data stored.

Keys used for cryptographic HSM functions are stored separate from other keys. This must be considered when importing keys.

This version of PinPad complies to the current ZKA specification 3.0. It supports loading and unloading against card account for both card types (Type 0 and Type 1) of the ZKA electronic purse. It also covers the necessary functionality for 'Loading against other legal tender'.

Key values are passed to the API as binary hexadecimal values, for example:
        0123456789ABCDEF = 0x01 0x23 0x45 0x67 0x89 0xAB 0xCD 0xEF

When hex values are passed to the API within strings, the hex digits 0xA to 0xF can be represented by characters in the ranges 'a' to 'f' or 'A' to 'F'.

The following commands and events were initially added to support the German ZKA standard, but may also be used for other national standards:

- WFS_INF_PIN_HSM_TDATA
- WFS_CMD_PIN_HSM_SET_TDATA
- WFS_CMD_PIN_SECURE_MSG_SEND

- WFS_CMD_PIN_SECURE_MSG_RECEIVE
- WFS_CMD_PIN_GET_JOURNAL
- WFS_SRVE_PIN_OPT_REQUIRED
- WFS_CMD_PIN_HSM_INIT
- WFS_SRVE_PIN_HSM_TDATA_CHANGED

2.5   Check Readers and Scanners (CHK)

Class Name   **CHK**

Class Identifier   WFS_SERVICE_CLASS_CHK = 5

This specification describes the functions provided by a Check Reader and Scanner (CHK) service.

Check image scanners are treated as a special case of check readers, i.e. image-enabled instances of the latter. This class includes devices with a range of features, from small hand-held read-only devices through which checks are manually swiped one at a time, to desktop units which automatically feed the check one at a time; recording the MICR data and check image, and endorse or encode the check.

In the U.S., checks are always encoded in magnetic ink for reading by Magnetic Ink Character Recognition (MICR), and a single font is always used. In Europe some countries use MICR and some use Optical Character Recognition (OCR) character sets, with different fonts, for their checks.

In all countries, typical fields found encoded on a check include the bank ID number and the account number. Part of the processing done by the bank is to also encode the amount on the check, usually done by having an operator enter the handwritten or typewritten face amount on a numeric keypad.

This service class is currently defined only for attended branch service.

2.6   Depository Units (DEP)

Class Name   DEP

Class Identifier   WFS_SERVICE_CLASS_DEP = 6

This specification describes the functions provided by a Depository (DEP) service.

A Depository is used for the acceptance and deposit of media into the device or terminal. There are two main types of depository: an envelope depository for the deposit of media in envelopes and a night safe depository for the deposit of bags containing bulk media.

An envelope depository accepts media, prints on the media and deposits the media into a holding container or bin. Some envelope depositories offer the capability to dispense an envelope to the customer at the start of a transaction. The customer takes this envelope, fills in the deposit media, possibly inscribes it and puts it into the deposit slot. The envelope is then accepted, printed and transported into a deposit container.

The envelope dispense mechanism may be part of the envelope depository device mechanism with the same entry/exit slot or it may be a separate mechanism with separate entry/exit slot.

Envelopes dispensed and not taken by the customer can be retracted back into the device. When the dispenser is a separate mechanism the envelope is retracted back into the dispenser container. When the dispenser is a common mechanism the envelope is retracted into the depository container.

A night safe depository normally only logs the deposit of a bag and does not print on the media.

2.7   Text Terminal Units (TTU)

Class Name   **TTU**

Class Identifier   WFS_SERVICE_CLASS_TTU = 7

This specification describes the functions provided by a Text Terminal Unit (TTU) service.

A Text Terminal Unit is a text i/o device, which applies both to ATM operator panels and to displays incorporated in devices such as PIN pads and printers. This service allows for the following categories of functions:

- Forms oriented input and output

- Direct display output

- Keyboard input

- LED settings and control

All position indexes are zero based, where column zero, row zero is the top-leftmost position.

If the device has no shift key, the WFS_CMD_TTU_READ_FORM and WFS_CMD_TTU_READ commands will return only upper case letters. If the device has a shift key, these commands return upper and lower case letters as governed by the user's use of the shift key.

2.8 Sensors and Indicators Units (SIU)

Class Name   **SIU**

Class Identifier   **WFS_SERVICE_CLASS_SIU = 8**

This specification describes the functions provided by a Sensors and Indicators Unit (SIU) service.

This service allows for the operation of the following categories of ports:

- Door sensors, such as cabinet, safe or vandal shield doors.

- Alarm sensors, such as tamper, seismic or heat sensors.

- Generic sensors, such as proximity or ambient light sensors.

- Key switch sensors, such as the ATM operator switch.

- Lamp/sign indicators, such as fascia light or audio indicators.

- Auxiliary indicators.

- Enhanced Audio Controller, for use by the partially sighted.

In self-service devices, the sensors and indicators unit is capable of dealing with external sensors, such as door switches, locks, alarms and proximity sensors, as well as external indicators, such as turning on lamps or heating.

2.9   Vendor Dependent Mode (VDM)

Class Name   VDM

Class Identifier   WFS_SERVICE_CLASS_VDM = 9

This specification describes the functions provided by a Vendor Dependent Mode (VDM) service.

In all device classes there needs to be some method of going into a vendor specific mode to allow for capabilities which go beyond the scope of the current XFS specifications. A typical usage of such a mode might be to handle some configuration or diagnostic type of function or perhaps perform some 'off-line' testing of the device. These functions are normally available on Self-Service devices in a mode traditionally referred to as Maintenance Mode or Supervisor Mode and usually require operator intervention. It is those vendor-specific functions not covered by (and not required to be covered by) XFS Service Providers that will be available once the device is in Vendor Dependent Mode.

This service provides the mechanism for switching to and from Vendor Dependent Mode. The VDM Service Provider can be seen as the central point through which all Enter and Exit VDM requests are synchronized.

Entry into, or exit from, Vendor Dependent Mode can be initiated either by an application or by the VDM Service Provider itself. If initiated by an application, then this application needs to issue the appropriate command to request entry or exit. If initiated by the VDM Service Provider i.e. some vendor dependent switch, then these request commands are in-appropriate and not issued.

Once the entry request has been made, all registered applications will be notified of the entry request by an event message. These applications must attempt to close all open sessions with XFS Service Providers as soon as possible and then issue an acknowledgement command to the VDM Service Provider when ready. Once all applications have acknowledged, the VDM Service Provider will issue event messages to these applications to indicate that the System is in Vendor Dependent Mode.

Similarly, once the exit request has been made all registered applications will be notified of the exit request by an event message. These applications must then issue an acknowledgement command to the VDM Service Provider immediately. Once all applications have acknowledged, the VDM Service Provider will issue event messages to these applications to indicate that the system has exited from Vendor Dependent Mode.

Thus, XFS compliant applications that do not need the system to be in Vendor Dependent Mode, must comply with the following:

- Every XFS application should open a session with the VDM Service Provider passing a valid ApplId and then register for all VDM entry and exit notices.

- Before opening a session with any other XFS Service Provider, check the status of the VDM Service Provider. If Vendor Dependent Mode is not "Inactive", do not open a session.

- When getting a VDM entry notice, close all open sessions with all XFS Service Providers as soon as possible and issue an acknowledgement for the entry to VDM.

- When getting a VDM exit notice, acknowledge at once.

- When getting a VDM exited notice, re-open any required sessions with other XFS Service Providers.

This is mandatory for self-service but optional for branch.

2.10 Cameras (CAM)

Class Name   **CAM**

Class Identifier   **WFS_SERVICE_CLASS_CAM = 10**

This specification describes the functions provided by a Camera (CAM) service.

Banking camera systems usually consist of a recorder, a video mixer and one or more cameras. If there are several cameras, each camera focuses a special place within the self-service area (e.g. the room, the customer or the cash tray). By using the video mixer it can be decided, which of the cameras should take the next photo. Furthermore data can be given to be inserted in the photo (e.g. date, time or bank code).

If there is only one camera that can switch to take photos from different positions, it is presented by the Service Provider as a set of cameras, one for each of its possible positions.

2.11 Alarms (ALM)

Class Name   **ALM**

Class Identifier   **WFS_SERVICE_CLASS_ALM = 11**

This specification describes the functions provided by an Alarm (ALM) service.

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.

2.12 Card Embossing Units (CEU)

Class Name   **CEU**

Class Identifier   WFS_SERVICE_CLASS_CEU = 12

This specification describes the functions provided by a Card Embossing Unit (CEU) service.

Embossing card units are generally viewed by XFS as compound devices with the following capabilities and features:

- Embossing of magnetic stripe card/ smart card.

- Reading/encoding magnetic stripe tracks 1, 2, and 3.

- Reading/writing smart card.

- LCD display/ keypad input.

The XFS services supporting the various embossing card unit components are outlined as follows:

- Embossing of magnetic stripe card/ smart card - Card Embossing Unit (CEU) service.

- Reading/encoding magnetic stripe tracks 1, 2, and 3 - ID Card (IDC) service, however when combined encoding/ embossing is performed the CEU service class is used.

- Reading/writing smart cards - ID Card (IDC) service, however when combined writing smart card/ embossing is performed the CEU service class is used.

- LCD display/ keypad input - Text Terminal Unit (TTU) service.

2.13 Cash-In Modules (CIM)

Class Name   **CIM**

Class Identifier   **WFS_SERVICE_CLASS_CIM = 13**

This specification describes the functions provided by a Cash-In Module (CIM) service.

Persistent values are maintained through power failures, open sessions, close session and system resets.

This specification covers the acceptance of items. An "item" is defined as any media that can be accepted and includes coupons, documents, bills and coins. However, if coins and bills are both to be accepted separate Service Providers must be implemented for each.

All currency parameters in this specification are expressed as a quantity of minimum dispense units, as defined in the description of the WFS_INF_CIM_CURRENCY_EXP command.

There are two types of CIM: Self-Service CIM and Teller CIM. A Self-Service CIM operates in an automated environment, while a Teller CIM has an operator present. The functionality provided by the following commands is only applicable to a Teller CIM:

WFS_CMD_CIM_SET_TELLER_INFO
WFS_INF_CIM_SET_TELLER_INFO

It is possible for the CIM to be part of a compound device with the Cash Dispenser Module (CDM). This CIM\CDM combination is referred to throughout this specification as a "cash recycler".

If the device is a cash recycler then, if cash unit exchanges are required on both interfaces, the exchanges cannot be performed concurrently. An exchange on one interface must be complete (the WFS_CMD_CIM_END_EXCHANGE must have completed) before an exchange can start on the other interface. The WFS_ERR_CIM_EXCHANGEACTIVE error code will be returned if the correct sequence is not adhered to.

The CIM interface can be used for all exchange operations on cash recycle devices, and this interface should be used for cash units of multiple currencies and/or denominations (including multiple note identifiers associated with the same denomination).

The event WFS_SRVE_CIM_COUNTS_CHANGED will be posted if an operation on the CDM interface affects the recycle cash unit counts which are available through the CIM interface.

The following commands on the CDM interface may affect the CIM counts:

       WFS_CMD_CDM_DISPENSE
       WFS_CMD_CDM_PRESENT
       WFS_CMD_CDM_RETRACT
       WFS_CMD_CDM_COUNT
       WFS_CMD_CDM_REJECT
       WFS_CMD_CDM_SET_CASH_UNIT_INFO
       WFS_CMD_CDM_END_EXCHANGE
       WFS_CMD_CDM_RESET
       WFS_CMD_CDM_TEST_CASH_UNITS

2.14 Card Dispensers (CRD)

Class Name   **CRD**

Class Identifier **WFS_SERVICE_CLASS_CRD = 14**

This specification describes the functions provided by a Card Dispenser (CRD) service.

A Card Dispenser is used to dispense a single card to a consumer from one or more bins. Most card dispensers also have the ability to retain a card to a bin.

2.15 Barcode Readers (BCR)

Class Name   **BCR**

Class Identifier   WFS_SERVICE_CLASS_BCR = 15

This specification describes the functions provided by a Barcode Reader (BCR) service.

Persistent values are maintained through power failures, open sessions, close session and system resets.

A Barcode Reader scans barcodes using any scanning technology. The device logic converts light signals or image recognition into application data and transmits it to the host system.

When an application wants to read a barcode, it issues a WFS_CMD_BCR_READ command to prepare the scanner to read any barcode presented to it. When a document is presented to the BCR and a barcode type is recognized, a completion event is received which contains the barcode data that has been read.

2.16 Item Processing Modules (IPM)

Class Name   IPM

Class Identifier   WFS_SERVICE_CLASS_IPM = 16

This specification describes the functions provided by an Item Processing Module (IPM) service.

This service class is currently defined only for self service devices.

In the U.S., checks are always encoded in magnetic ink for reading by Magnetic Ink Character Recognition (MICR), and a single font is always used. In Europe some countries use MICR and some use Optical Character Recognition (OCR) character sets, with different fonts, for their checks.

Item Processing Modules accept one or more media items (Checks, Giros, etc) and process these items according to application requirements. The IPM class supports devices that can handle a single item as well as those devices that can handle bunches of items. The following are the three principle device types:

- Single Item: can accept and process a single item at a time.

- Multi-Item Feed with no stacker (known as an escrow in some environments): can accept a bunch of media from the customer but each item has to be processed fully (i.e. deposited in a bin or returned) before the next item can be processed.

- Multi-Item Feed with a stacker: can accept a bunch of media from the customer and all items can be processed together.

The IPM class provides applications with an interface to control the following functions (depending on the capabilities of the specific underlying device):

- Capture an image of the front of an item in multiple formats and bit depths.

- Capture an image of the back of an item in multiple formats and bit depths.

- Read the code line of an item using MICR reader.

- Read the code line of an item using OCR.

- Endorse (print text) on an item.

- Stamp an item.

- Return an item to the customer.

- Deposit an item in a bin.

- Retract items left by the customer.

The IPM device class uses the concept of a Media-In transaction to track and control a customer's interaction with the device. A Media-In transaction consists of one or more WFS_CMD_IPM_MEDIA_IN commands. The transaction is initiated by the first WFS_CMD_IPM_MEDIA_IN command and remains active until the transaction is either confirmed through WFS_CMD_IPM_MEDIA_IN_END, or terminated by WFS_CMD_IPM_MEDIA_IN_ROLLBACK, WFS_CMD_IPM_RETRACT_MEDIA or WFS_CMD_IPM_RESET. While a transaction is active the WFS_INF_IPM_TRANSACTION_STATUS command reports the status of the current transaction. When a transaction is not active the WFS_INF_IPM_TRANSACTION_STATUS command reports the status of the last transaction.

There are primarily two types of devices supported by the IPM, those devices with a stacker and those without.

3. Planned Enhancements and Extensions

XFS currently includes specifications for access to the key classes of financial peripherals, as listed above. These existing specifications will be extended and enhanced based on vendor and user experience with them. In addition, new device classes that customers and vendors request may be evaluated for inclusion in future versions of this specification.

Please submit comments and questions to

xfs-helpdesk@cenorm.be

Or contact

Luc Van den Berghe
CEN Workshop Manager
Rue de Stassart 36
B-1050 Brussels
Luc.vandenberghe@cenorm.be
Tel: + 32 2 55 00 957