CEN

WORKSHOP

AGREEMENT

CWA 13449-5

December 1998

ICS 35.200;35.240.40

English version

Extensions for Financial Services (XFS) interface specification - Part 5: Cash Dispenser Device Class Interface - Programmer's Interface

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 Central Secretariat 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, Czech Republic, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and United Kingdom.

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

Central Secretariat: rue de Stassart, 36 B-1050 Brussels

Ref. No. CWA 13449-5:1998 E

Contents

Foreword

This CWA is revision 2.0 of the XFS interface specification. Release 2.0 extends the scope of the XFS interface specification to include both the self service/ATM environment as well as the branch environment. The new specification now fully supports cameras, deposit units, identification cards, PIN pads, sensors and indicator units, text terminals, cash dispenser modules and a wide variety of printing mechanisms.

This specification was originally developed by the Banking Solutions Vendor Council (BSVC), and is endorsed by the CEN/ISSS Workshop on XFS. This Workshop gathers both suppliers (among others the BSVC members) 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.

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 2.00.

This CWA is supplemented by a set of release notes, which are available from the CEN/ISSS Secretariat (an on-line version of these release notes is available from http://www.cenorm.be/isss/Workshop/XFS/release-notes.htm).

0. Introduction

This is part 5 of the multi-part CWA 13449, describing Release 2.0 of the XFS interface specification.

The full CWA 13449 "Extensions for Financial Services (XFS) interface specification"consists of the following parts:

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

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 from the CEN/ISSS Secretariat (contact isss@cenorm.be or download from http://www.cenorm.be/isss/ Workshop/XFS/release-notes.htm).

The information in this document originally contributed by members of the Banking Solutions Vendor Council and endorsed by the CEN/ISSS Workshop on XFS, 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.

The XFS specifications are now further developed in the CEN/ISSS Workshop on XFS. CEN/ISSS Workshops are open to all interested parties offering to contribute. Parties interested in participating should contact the CEN/ISSS Secretariat (isss@cenorm.be).

A Software Development Kit (SDK) which supplies the components and tools to allow the implementation of compliant applications and services is available from Microsoft[1].

To the extent that date processing occurs, all XFS Workshop participants agree that the XFS specifications are Year 2000 compliant.

Revision History:

1.0	**May 24, 1993**	**Initial release of API and SPI specification**
1.11	**February 3, 1995**	**Separation of specification into separate documents for API/SPI and service class definitions, with updates**
2.00	**November 11, 1996**	**Updated release encompassing self-service environment.**
	October 6, 1998	**WOSA/XFS Release 2.00 as originally developed by the BSVC, has been formally accepted as a CEN Workshop Agreement by the CEN/ISSS XFS Workshop and the name WOSA/XFS has been changed into XFS. In spite of the name change, certain occurrencies of WOSA/XFS however still appear in the documentation, for compatibility reasons**

[1] Microsoft is a registered trademark, and Windows and Windows NT are trademarks of Microsoft Corporation

1. 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 the Extensions for Financial Services specification is to standardize command 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 the union of the sets of 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 command set defined for the class.

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 bills, 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. Cash Dispensers

This specification describes the functionality of the services provided by cash dispenser module (CDM) services 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 a cash dispenser module (CDM) service that applies to both Automated Teller Safes (ATS) and Automated Teller Machines (ATM).

The components of an Automated Teller Safe are a cash (bills) dispenser, a transport unit, an output unit, and in some cases it also contains a coin dispenser and an alert unit.

An Automated Teller Machine contains the modules for cash dispensing plus additional modules such as magnetic card reader/writer, PIN pad, terminal, etc. The modules used for cash dispensing are essentially the same as those contained in the ATS: a bill/coupon/document dispenser, a transport module, an output module, and a coin dispenser, if available. Therefore, the cash dispensing functionality of the ATS and of the ATM is included in a single service class definition, referred to in this specification as “CDM” (cash dispenser module).

The implementation of the individual commands is device dependent (ATS or ATM). This is described in the documentation of each function.

The following table specifies which **WFSGetInfo, WFSAsyncGetInfo, WFSExecute** and **WFSAsyncExecute** commands are typically meaningful for the two kinds of devices.

Command	ATS	ATM
WFS_INF_CDM_STATUS	X	X
WFS_INF_CDM_CAPABILITIES	X	X
WFS_INF_CDM_CASH_UNIT_INFO	X	X
WFS_INF_CDM_TELLER_INFO	X	
WFS_INF_CDM_TELLER_POSITIONS	X	
WFS_INF_CDM_CURRENCY_EXP	X	X
WFS_INF_CDM_MIX_TYPES	X	X
WFS_INF_CDM_MIX_TABLE	X	X
WFS_INF_CDM_PRESENT_STATUS		X
WFS_CMD_CDM_DENOMINATE	X	X
WFS_CMD_CDM_DISPENSE	X	X
WFS_CMD_CDM_PRESENT		X
WFS_CMD_CDM_REJECT		X
WFS_CMD_CDM_RETRACT		X
WFS_CMD_CDM_CASH_IN	X	
WFS_CMD_CDM_SET_TELLER_INFO	X	
WFS_CMD_CDM_SET_CASH_UNIT_INFO	X	X
WFS_CMD_CDM_START_EXCHANGE	X	X
WFS_CMD_CDM_END_EXCHANGE	X	X
WFS_CMD_CDM_OPEN_SAFE DOOR	X	X
WFS_CMD_CDM_CLOSE_SHUTTER		X
WFS_CMD_CDM_OPEN_SHUTTER		X
WFS_CMD_CDM_CHECK_VANDALISM		X
WFS_CMD_CDM_CALIBRATE_CASH_UNIT	X	X
WFS_CMD_CDM_SET_TELLER_POSITIONS	X	
WFS_CMD_CDM_CASH_IN_START	X	
WFS_CMD_CDM_CASH_IN_END	X	
WFS_CMD_CDM_CASH_IN_ROLLBACK	X	
WFS_CMD_CDM_SET_MIX_TABLE	X	X

NOTE: All money amount parameters in this specification are expressed as a quantity of minimum dispense units for the respective currency, as defined in the description of the WFS_INF_CDM_CURRENCY_EXP command (Section 3.6).

3. Info Commands

3.1 WFS_INF_CDM_STATUS

Description This command is used to request the status of the devices attached to a CDM, such as the safedoor and other devices, or to retrieve device-specific information.

Input Param None.

Output Param

```
LPWFSCDMSTATUS      lpStatus;

typedef struct _wfs_cdm_status
    {
    WORD              fwDevice;
    WORD              fwSafeDoor;
    WORD              fwDispenser;
    WORD              fwIntermediateStacker;
    LPWFSCDMOUTPOS *  lppOutputPositions;
    LPSTR             lpszExtra;
    } WFSCDMSTATUS, *  LPWFSCDMSTATUS;
```

fwDevice
Specifies the state of the cash dispenser device as one of the following flags:

Value	Meaning
WFS_CDM_DEVONLINE	The device is online. This is returned when the dispenser is present and operational. The *fwDispenser* field should be checked to determine the state of the cash units; if the state is WFS_CDM_DISPCUSTOP, no bills can be dispensed. The *fwSafeDoor* field can be checked to determine the state of the safe door, but note that the effect of the door state on the ability to dispense cash depends on the implementation of the specific service.
WFS_CDM_DEVOFFLINE	The device is present and powered on, but is offline.
WFS_CDM_DEVPOWEROFF	The device is present but is powered off.
WFS_CDM_DEVBUSY	The device is busy processing a **WFS[Async]Execute** or **WFS[Async]GetInfo** request.
WFS_CDM_DEVNODEVICE	There is no device connected.
WFS_CDM_DEVUSERERROR	The device is present but a person is preventing proper device operation. The application should suspend 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_CDM_DEVONLINE) or a permanent error condition has occurred (WFS_CDM_DEVHWERROR).
WFS_CDM_DEVHWERROR	The device is present and powered on, but is inoperable due to a hardware error. The device stays in this condition until the hardware error is cleared.

fwSafeDoor
Specifies the state of the safe door as one of the following flags:

Value	Meaning
WFS_CDM_DOORNOTSUPPORTED	Physical device has no safe door or door state reporting is not supported.
WFS_CDM_DOOROPEN	Safe door is open.
WFS_CDM_DOORCLOSED	Safe door is closed but not locked.
WFS_CDM_DOORLOCKED	Safe door is closed and locked.
WFS_CDM_DOORUNKNOWN	Due to a hardware error or other condition, the state of the door cannot be determined.

fwDispenser
Specifies the state of the dispenser cash units as one of the following flags:

Value	Meaning
WFS_CDM_DISPOK	All logical cash units present are in a good state.
WFS_CDM_DISPCUSTATE	One of the logical cash units present is in an abnormal state. The dispenser is operational, but one or more of the cash units is in a low, empty or inoperative condition. Bills can still be dispensed from at least one of the cash units.
WFS_CDM_DISPCUSTOP	Due to a cash unit failure dispensing is impossible. The dispenser is operational, but no bills can be dispensed because all of the cash units are in an empty or inoperative condition. This state also occurs when a reject cash unit is full or no reject cassette is present.
WFS_CDM_DISPCUUNKNOWN	Due to a hardware error or other condition, the state of the cash units cannot be determined.

fwIntermediateStacker
Specifies the state of the intermediate stacker as one of the following flags:

Value	Meaning
WFS_CDM_ISEMPTY	The intermediate stacker is empty.
WFS_CDM_ISNOTEMPTY	The intermediate stacker is not empty.
WFS_CDM_ISUNKNOWN	Due to a hardware error or other condition, the state of the intermediate stacker cannot be determined.
WFS_CDM_ISNOTSUPPORTED	The physical device has no intermediate stacker.

lppOutputPositions
Pointer to a NULL terminated array of pointers to WFSCDMOUTPOS structures (one for each supported output position):

```
typedef struct _wfs_cdm_outpos
    {
    WORD          fwPosition;
    WORD          fwShutter;
    WORD          fwOutputPosition;
    WORD          fwTransport;
    } WFSCDMOUTPOS, * LPWFSCDMOUTPOS;
```

fwPosition
Specifies the output position as one of the following flags:

Value	Meaning
WFS_CDM_POSLEFT	left output position.
WFS_CDM_POSRIGHT	right output position.
WFS_CDM_POSCENTER	center output position.

fwShutter
Specifies the state of the shutter as one of the following flags:

Value	Meaning
WFS_CDM_SHTCLOSED	The shutter is closed.
WFS_CDM_SHTOPEN	The shutter is opened.
WFS_CDM_SHTJAMMED	The shutter is jammed.
WFS_CDM_SHTUNKNOWN	Due to a hardware error or other condition, the state of the shutter cannot be determined.
WFS_CDM_SHTNOTSUPPORTED	The physical device has no shutter or shutter state reporting is not supported.

fwOutputPosition
Specifies the state of the cash tray as one of the following flags:

Value	Meaning
WFS_CDM_CTEMPTY	The cash tray is empty.
WFS_CDM_CTNOTEMPTY	The cash tray is not empty.
WFS_CDM_CTUNKNOWN	Due to a hardware error or other condition, the state of the cash tray cannot be determined.
WFS_CDM_CTNOTSUPPORTED	The physical device has no cash tray or cash tray state reporting is not supported.

fwTransport
Specifies the state of the transport mechanism as one of the following flags:

Value	Meaning
WFS_CDM_TPOK	The transport is in a good state.
WFS_CDM_TPINOP	The transport is inoperative due to a hardware failure or media jam.
WFS_CDM_TPUNKNOWN	Due to a hardware error or other condition, the state of the transport cannot be determined.
WFS_CDM_TPNOTSUPPORTED	The physical device has no transport or transport state reporting is not supported.

lpszExtra
Points 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 will be null-terminated, with the final string terminating with two null characters.

Error Codes There are no additional error codes 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.

3.2 WFS_INF_CDM_CAPABILITIES

Description This command is used to retrieve the capabilities of the cash dispenser.

Input Param None.

Output Param

```
LPWFSCDMCAPS  lpCaps;

typedef struct _wfs_cdm_caps
   {
   WORD          wClass;
   WORD          fwType;
   WORD          wMaxBills;
   WORD          wMaxCoins;
   BOOL          bCompound;
   BOOL          bShutter;
   BOOL          bRetract;
   BOOL          bSafeDoor;
   BOOL          bCoins;
   BOOL          bCylinders;
   BOOL          bCashBox;
   BOOL          bCashIn;
   BOOL          bRefill;
   BOOL          bAutoDeposit;
   BOOL          bVandalCheck;
   BOOL          bIntermediateStacker;
   BOOL          bBillsTakenSensor;
   WORD          fwOutputPositions;
   LPSTR         lpszExtra;
   } WFSCDMCAPS, * LPWFSCDMCAPS;
```

wClass
Specifies the logical service class. Value is:
WFS_SERVICE_CLASS_CDM

fwType
Specifies the type of the physical device driven by the logical service. Values are:

Value	Meaning
WFS_CDM_TYPEATSAFE	Device is an Automated Teller Safe.
WFS_CDM_TYPEATMACHINE	Device is an Automated Teller Machine.

wMaxBills
Specifies the maximum number of bills or documents that can be dispensed by a single dispensing command; normally reflects hardware limitations of the device. If *wMaxBills* is 0

then no limit applies to the number of bills to be dispensed in a single dispense command. If the number of bills to be dispensed exceeds the hardware limitations of the device, this will be implemented by the Service Provider by multiple dispenses.

wMaxCoins
Specifies the maximum number of coins that can be dispensed by a single dispensing command; normally reflects hardware limitations of the device.

bCompound
Specifies whether the logical device is part of a compound physical device and is either TRUE or FALSE.

bShutter
Specifies whether the cash dispenser transport mechanism includes a shutter which normally is controlled by the DISPENSE or PRESENT command, but can be explicitly opened using WFS_CMD_CDM_OPEN_SHUTTER and closed using WFS_CMD_CDM_CLOSE_SHUTTER. Value is either TRUE or FALSE.

bRetract
Specifies whether the cash dispenser transport mechanism has the ability to retract presented bills. Value is either TRUE or FALSE. If no separate retract cash unit is present in the physical device, then retracts are retracted to the reject cash unit.

bSafedoor
Specifies whether the safe door has an electrical time lock, controlled by the WFS_CMD_CDM_OPEN_SAFE_DOOR command. Value is either TRUE or FALSE.

bCoins
Specifies whether the cash dispenser device includes a coin dispensing facility. Value is either TRUE or FALSE.

bCylinders
Specifies whether the coin dispenser device allows for number of coins per cylinder as input, or whether only totals are allowed. Value is either TRUE or FALSE.

bCashBox
Specifies whether the service provides the ability to count for a cashbox assigned to each teller. Value is either TRUE or FALSE. This field is always FALSE if the *fwType* is WFS_CDM_TYPEATMACHINE.

bCashIn
Specifies whether the service provides the ability to accumulate a cash in amount per currency assigned to each teller. Value is either TRUE or FALSE.

bRefill
Specifies that the device is equipped with cassettes that can be refilled during runtime by just dropping a bundle of bills on top of the stack. Normally, the device cannot detect and count the number of bills in the cassettes. Value is either TRUE or FALSE.

bAutoDeposit
Specifies whether the cash in device is able to deposit money and to provide the denomination as result. Value is either TRUE or FALSE.

bVandalCheck
Specifies whether the physical device includes a feature to check for vandalism. Value is either TRUE or FALSE.

bIntermediateStacker
Specifies whether or not the cash dispenser device supports stacking bills to an intermediate position before the bills are presented to the exit position. Value is either TRUE or FALSE. TRUE means the capability exists and the DISPENSE command can optionally stack bills using the input parameter *bPresent*=FALSE. Following this DISPENSE, a PRESENT command or REJECT command may be issued. FALSE means the DISPENSE command with a *bPresent* parameter of FALSE will be rejected with WFS_ERR_UNSUPP_COMMAND.

bBillsTakenSensor
Specifies whether or not the cash dispenser has the ability to detect when bills delivered to the

exit position are taken by the user. TRUE means a sensor exists and the “bills taken” condition can be detected.

fwOutputPositions
Specifies which output positions are available on the physical device as a combination of the following flags:

Value	Meaning
WFS_CDM_POSLEFT	Device has a left output position.
WFS_CDM_POSRIGHT	Device has a right output position.
WFS_CDM_POSCENTER	Device has a center output position.

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.

Error Codes There are no additional error codes 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.

3.3 WFS_INF_CDM_CASH_UNIT_INFO

Description This command is used to get information about the status and contents of the logical and physical cash units used in the dispenser module.

Each logical bill or coin type cash unit can comprise one or more physical cash units. For all other types of cash unit there is a one to one relationship between physical and logical cash unit.

Information on logical cash units is used by the application to process transactions while the information on physical cash units is used during replenishment and maintenance operations.

All counters returned by this command are software counters and therefore might not represent the actual physical cash counts.

Input Param None.

Output Param
```
LPWFSCDMCUINFO      lpCashUnitInfo;

typedef struct _wfs_cdm_cu_info
  {
  USHORT              usTellerID;
  USHORT              usCount;
  LPWFSCDMCASHUNIT *  lppList;
  } WFSCDMCUINFO, * LPWFSCDMCUINFO;
```

usTellerID
This field is not used in this command and is always NULL. In other commands that use this structure, and that relate to individual tellers (i.e., WFS_CMD_CDM_SET_CASH_UNIT_INFO, WFS_CMD_CDM_START_EXCHANGE, WFS_CMD_CDM_END_EXCHANGE), this field contains the appropriate teller ID value.

usCount
Specifies the number of cash unit structures returned.

lppList
Pointer to an array of pointers to cash unit structures:

```
typedef struct _wfs_cdm_cashunit
   {
   USHORT          usNumber;
   USHORT          usType;
   CHAR            cUnitID[5];
   CHAR            cCurrencyID[3];
   ULONG           ulValues;
   ULONG           ulInitialCount;
   ULONG           ulCount;
```

```
ULONG           ulMinimum;
ULONG           ulMaximum;
BOOL            bAppLock;
BOOL            bDevLock;
USHORT          usStatus;
LPSTR           lpPhysicalPositionName;
USHORT          usNumPhysicalCUs;
LPWFSCDMPHCU    *lppPhysical;
} WFSCDMCASHUNIT, * LPWFSCDMCASHUNIT;
```

usNumber
Logical number of cash unit. Each structure has a unique logical number starting with a value of one (1) for the first structure, and incrementing by one for each subsequent structure.

usType
Type of cash unit. Possible values are:

Value	Meaning
WFS_CDM_TYPENA	Not applicable; typically means cash unit is missing.
WFS_CDM_TYPEREJECTCASSETTE	Reject cassette of the cash dispenser.
WFS_CDM_TYPEBILLCASSETTE	Bill cassette of the cash dispenser.
WFS_CDM_TYPECOINCYLINDER	Cylinder of the coin dispenser.
WFS_CDM_TYPECOINDISPENSER	Coin dispenser as a whole unit.
WFS_CDM_TYPERETRACTCASSETTE	Retract cassette of the cash dispenser.
WFS_CDM_TYPECOUPON	Cassette for coupons or advertising material.
WFS_CDM_TYPEDOCUMENT	Cassette for documents.

cUnitID
Cash unit ID. This field provides information identifying a cash unit.

cCurrencyID
Currency ID (ISO format). Three blanks for a reject or a retract cassette which stores more than one currency type and for coupon cassettes which do not contain currencies.

ulValues
Values of coins/bills. This is the value of a single coin or bill in the cash unit, not the total value of the coins or bills present. (Amount expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP.) Zero for a reject or a retract cassette and for coupon cassettes which do not contain valuable documents.

ulInitialCount
Initial number of coins/bills. This value is persistent; i.e, maintained through power failures, opens, closes and system resets. It is set by the WFS_CMD_CDM_END_EXCHANGE and the WFS_CMD_CDM_SET_CASH_UNIT_INFO commands.

ulCount
Actual count of coins/bills. This value is persistent; i.e, maintained through power failures, opens, closes and system resets. It is set by the WFS_CMD_CDM_END_EXCHANGE and the WFS_CMD_CDM_SET_CASH_UNIT_INFO commands, and is adjusted each time coins/bills are moved from or to the cash unit by a **WFSExecute** command. Note that for a reject cassette, this value is unreliable, since the typical reason for dumping bills to the reject cassette is a suspected count failure. For a retract cassette this value specifies the number of retracts.

ulMinimum
Specifies the minimum number of bills in a cash unit. If the number falls short of this value a threshold event (WFS_USRE_CDM_CASHUNITTHRESHOLD) is created. For retract and reject cassettes this value equals zero.

ulMaximum
Specifies the maximum number of bills in a cash unit. This value is needed for cash units that are refilled during a cash in operation, reject and retract cassettes. For all other cash

units it equals zero. If the number exceeds this value a threshold event (WFS_USRE_CDM_CASHUNITTHRESHOLD) is created.

bAppLock
Application lock status of the logical cash unit. If set to TRUE, the cash unit is locked by the application and cannot be used until unlocked. Application locks can only be applied to bill cassettes and coin cylinders.

bDevLock
Device lock status of the logical cash unit. If set to TRUE, the cash unit is locked by the device and cannot be used until unlocked. Device locks can only be applied to bill cassettes and coin cylinders. Support for device locks is vendor-dependent.

usStatus
Describes the status of the logical cash unit. Available values are:

Value	Meaning
WFS_CDM_STATCUOK	The cash unit is in a good state.
WFS_CDM_STATCUFULL	The reject or retract cassette is full.
WFS_CDM_STATCUHIGH	The reject or retract cassette is almost full (threshold).
WFS_CDM_STATCULOW	The cassette or coin cylinder is almost empty (threshold).
WFS_CDM_STATCUEMPTY	The cassette or coin cylinder is empty.
WFS_CDM_STATCUINOP	The cassette or coin cylinder is inoperative.
WFS_CDM_STATCUMISSING	The cassette, reject or retract cassette is missing.
WFS_CDM_STATCUNOVAL	The values of the specified cash unit are not available. This can be the case when the cassette is changed without using the operator functions.
WFS_CDM_STATCUNOREF	There is no reference value available for the notes in this cassette.

lpPhysicalPositionName
This field points to a name (e.g. “hopper 1”) indicating the physical location in the dispenser device in which this cash unit is installed. This field is only used when the logical cash unit refers to one physical cash unit.

usNumPhysicalCUs
This value indicates the number of physical cash unit structures returned. If this field is zero then the logical cash unit structure above relates directly to a physical cash unit.

lppPhysical
Pointer to an array of pointers to physical cash unit structures:

```
typedef struct _wfs_cdm_physicalcu
   {
   LPSTR     lpPhysicalPositionName;
   CHAR      cUnitID[5];
   ULONG     ulCount;
   USHORT    usPStatus;
   } WFSCDMPHCU, * LPWFSCDMPHCU;
```

lpPhysicalPositionName
This field points to a name (e.g. “hopper 1”) indicating the physical location in the dispenser device in which this cash unit is installed.

cUnitID
Cash unit ID. This field provides information identifying a physical cassette clearly.

ulCount
Actual count of coins/bills in the physical cash unit.

usPStatus
Describes the status of the physical cash unit. Same values as *usStatus*.

Error Codes There are no additional error codes generated by this command.

Comments None.

3.4 WFS_INF_CDM_TELLER_INFO

Description This command is used for getting the tallies assigned to a teller. A set of tallies can be requested for each currency assigned to the teller.

Input Param

```
LPWFSCDMTELLERINFO  lpTellerInfo;

typedef struct _wfs_cdm_teller_info
   {
   USHORT       usTellerID;
   CHAR         cCurrencyID[3];
   } WFSCDMTELLERINFO, *LPWFSCDMTELLERINFO;
```

usTellerID
Identification of teller. The service provider maintains a list of valid teller IDs; if the *usTellerID* value is not present in the list, an error is reported. The implementation of the list is vendor dependent.

cCurrencyID
Identification of currency (ISO format).

Output Param

```
LPWFSCDMTELLERTOTALS       lpTellerTotals;

typedef struct _wfs_cdm_teller_totals
   {
   USHORT       usTellerID;
   CHAR         cCurrencyID[3];
   ULONG        ulBills;
   ULONG        ulCoins;
   ULONG        ulCashIn;
   ULONG        ulCashBox;
   } WFSCDMTELLERTOTALS, * LPWFSCDMTELLERTOTALS;
```

usTellerID
Identification of teller.

cCurrencyID
Identification of currency (ISO format).

ulBills
Amount of money dispensed from bill cassettes. (Amount expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP.)

ulCoins
Amount of money dispensed from coin cylinders. (Amount expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP.)

ulCashIn
Amount of money cashed in by teller. (Amount expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP.)

ulCashBox
Amount dispensed from teller's cash box. (Amount expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP.)

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_INVALIDCURRENCY	Specified currency not currently available
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list.

Comments None.

3.5 WFS_INF_CDM_TELLER_POSITIONS

Description This command is used to get the output position assigned to each teller.

Input Param None.

Output Param `LPWFSCDMTELLERPOS * lppTellerPos;`
Pointer to a null-terminated array of pointers to teller position structures:

```
typedef struct _wfs_cdm_teller_pos
   {
   USHORT    usTellerID;
   USHORT    usPosition;
   } WFSCDMTELLERPOS, *LPWFSCDMTELLERPOS;
```

usTellerID
Identification of teller.

usPosition
Output position where cash is presented to the teller. Possible values are:

Value	Meaning
WFS_CDM_POSNULL	No position assigned to teller.
WFS_CDM_POSLEFT	Left position assigned to teller.
WFS_CDM_POSRIGHT	Right position assigned to teller.
WFS_CDM_POSCENTER	Center position assigned to teller.

Error Codes There are no additional error codes generated by this command.

Comments None.

3.6 WFS_INF_CDM_CURRENCY_EXP

Description This command is used for getting the exponents assigned to each currency used.

Input Param None.

Output Param `LPWFSCDMCURRENCYEXP *  lppCurrencyExp;`
Pointer to a null-terminated array of pointers to currency exponent structures:

```
typedef struct _wfs_cdm_currency_exp
   {
   CHAR          cCurrencyID[3];
   SHORT         sExponent;
   } WFSCDMCURRENCYEXP, *LPWFSCDMCURRENCYEXP;
```

cCurrencyID
Currency identifier (ISO 4217 format).

sExponent
Currency exponent.

Error Codes There are no additional error codes generated by this command.

Comments For each currency ISO 4217 defines the currency identifier (a three character code) and a currency unit (e.g., German mark, Italian lira). In the interface defined by this specification, every money amount is specified in terms of multiples of the minimum dispense unit, which is equal to the currency unit times the currency exponent. Thus a money amount parameter relates to the actual cash amount as follows:

<cash_amount> = <money_amount_parameter> * 10^<sExponent>

Example #1 — Germany
Currency identifier is 'DEM'
Currency unit is 1 German mark (= 100 pfennig)
A service provider is developed for an ATM that can dispense coins down to one pfennig. The currency exponent (*sExponent*) is set to -2 (minus two), so the minimum dispense unit is one

pfennig (1 * 10^-2 mark); all amounts at the XFS interface are in pfennigs. Thus a money amount parameter of 10050 is 100 marks and 50 pfennig.

Example #2 — Italy

Currency identifier is 'LIT'

Currency unit is 1 Italian lira

A service provider is required to dispense a minimum amount of 100 lire. The currency exponent (*sExponent*) is set to +2 (plus two), so the minimum dispense unit is 100 lire; all amounts at the XFS interface are in multiples of 100 lire. Thus a money amount parameter of 150 is 15000 lire.

3.7 WFS_INF_CDM_MIX_TYPES

Description This command is used to retrieve a list of supported mix algorithms and available house mix tables.

Input Param None.

Output Param `LPWFSCDMMIXTYPE * lppMixTypes;`
Pointer to a null-terminated array of pointers to mix type structures:

```
typedef struct _wfs_cdm_mix_type
   {
   USHORT       usMixNumber;
   USHORT       usMixType;
   USHORT       usSubType;
   LPSTR        lpszName;
   } WFSCDMMIXTYPE, *LPWFSCDMMIXTYPE;
```

usMixNumber
Internal number identifying the mix algorithm or the house mix table. This number is passed to the WFS_INF_CDM_MIX_TABLE, WFS_CMD_CDM_DISPENSE and WFS_CMD_CDM_DENOMINATE commands.

usMixType
Specifies whether the mix type is an algorithm or a house mix table. Values are:

Value	Meaning
WFS_CDM_MIXALGORITHM	Denominations are calculated using a mix algorithm.
WFS_CDM_MIXTABLE	Denominations are calculated using a house mix table.

usSubType
Contains a service provider defined number that identify the type of algorithm or table. Predefined values are:

Value	Meaning
WFS_CDM_MIX_MINIMUM_NUMBER_OF_BILLS	The minimum number of bills possible is taken.
WFS_CDM_MIX_EQUAL_EMPTYING_OF_CASH_UNITS	The cash units are equally emptied.

lpszName
Points to the name of the table/algorithm used.

Error Codes There are no additional error codes generated by this command.

Comments None.

3.8 WFS_INF_CDM_MIX_TABLE

Description This command is used to retrieve the house mix table specified by the mix number.

Input Param `LPUSHORT lpusMixNumber;`

lpusMixNumber
Points to the number of the requested house mix table.

Output Param LPWFSCDMMIXTABLE lpMixTable;

```
typedef struct _wfs_cdm_mix_table
   {
   USHORT           usMixNumber;
   LPSTR            lpszName;
   USHORT           usRows;
   USHORT           usCols;
   LPULONG          lpulMixHeader;
   LPWFSCDMMIXROW * lppMixRows;
   } WFSCDMMIXTABLE, *LPWFSCDMMIXTABLE;
```

usMixNumber
Internal number of house mix table.

lpszName
Points to the name of the table.

usRows
Number of rows in the house mix table. There is at least one row for each distinct total amount to be denominated.

usCols
Number of columns in the house mix table. There is one column for each distinct bill and coin value included in the mix.

lpulMixHeader
Pointer to an array of length *usCols* of unsigned longs ; each element defines the value of the bill or coin corresponding to its respective column. (See WFS_INF_CDM_CURRENCY_EXP.)

lppMixRows
Pointer to an array (of length *usRows*) of pointers to WFSCDMMIXROW structures:

```
typedef struct _wfs_cdm_mix_row
   {
   ULONG              ulAmount;
   LPUSHORT           lpusMixture;
   } WFSCDMMIXROW, *LPWFSCDMMIXROW;
```

ulAmount
Amount denominated by this mix row (See WFS_INF_CDM_CURRENCY_EXP).

lpusMixture
Pointer to a mix row, an array of length *usCols* of unsigned integers; each element defines the quantity of each bill and coin denomination in the mix used in the denomination of *ulAmount*.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_INVALIDMIXNUMBER	The *lpusMixNumber* parameter does not correspond to a defined mix table.

Comments None.

3.9 WFS_INF_CDM_PRESENT_STATUS

Description This command is used to inform about the status of the last dispense and is used only by ATM's. It is necessary to decide whether the money was in customer access or not. After a reboot this command returns the status of the last dispense before reboot. This status is valid until the next dispense command.

Input Param None.

Output Param LPWFSCDMPRESENTSTATUS lpPresentStatus;

```
typedef struct _wfs_cdm_present_status
  {
```

```
LPWFSCDMDENOMINATION   lpDenomination;
WORD                   wPresentState;
LPSTR                  lpszExtra;
} WFSCDMPRESENTSTATUS, *LPWFSCDMPRESENTSTATUS;
```

lpDenomination
Pointer to a WFSCDMDENOMINATION structure, describing the denominations used for the dispense operation. For a description of the WFSCDMDENOMINATION structure see the definition of the command WFS_CMD_CDM_DENOMINATE.

wPresentState
State of the money. Possible values are:

Value	Meaning
WFS_CDM_PRESENTED	The money was presented. This value is set as soon as bills are accessible by the customer.
WFS_CDM_NOTPRESENTED	The money was not presented.
WFS_CDM_UNKNOWN	It is unknown if the money could be accessed by the customer.

lpszExtra
Points 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 will be null-terminated, with the final string terminating with two null characters.

Error Codes There are no additional error codes generated by this command.

Comments None.

4. Execute Commands

4.1 WFS_CMD_CDM_DENOMINATE

Description

This command, which is designed to support denomination dialogues, will provide a denomination, i.e., a mix of bills and/or coins, capable of being paid out according to the amount (in terms of the minimum dispense unit) and currency specified, the mix algorithm selected and the desired denomination. In addition to that, it provides a facility for checking any given denomination for its capability of being paid out.

For the denomination of a specified amount the money can be retrieved from three different sources:

- the cash dispenser
- the coin dispenser (see WFS_INF_CDM_CAPABILITIES)
- the teller's cash box (see WFS_INF_CDM_CAPABILITIES)

The configuration specifies which of these three sources are allowed to be used in the denomination. For a ATS all three can be used. If the device used is an ATM, only the cash dispenser and, optionally, the coin dispenser can be available.

For the cash dispenser module there is a parameter (*wMaxBills* in the WFS_INF_CDM_CAPABILITIES command) controlling the maximum number of bills or documents that can be paid out within a single dispensing command. The coin dispenser has a parameter (*wMaxCoins* in the WFS_INF_CDM_CAPABILITIES command) specifying the maximum number of coins that can be paid out.

Existing variants for dispensing of bills or documents are:

- House mix tables (denomination tables), defined in the configuration
- Denomination algorithms, such as balanced use of the different cash units, use of bills with highest possible values, etc.

There are four distinct combinations of the inputs to this command:

1. Input parameters are currency and denomination, with mix number WFS_CDM_INDIVIDUAL and amount equal to zero. In this case the service checks only whether the denomination is valid according to the counters and the states of the cash units.
2. Input parameters are currency, amount and denomination, with mix number WFS_CDM_INDIVIDUAL. The service checks amount and denomination and returns amount, currency and denomination, if the denomination specified is OK.
3. Input parameters are currency, amount and mix number. The denomination is performed depending on the specified amount and mix number, and the configuration (coin dispenser and/or cash box; see WFS_INF_CDM_CAPABILITIES). The service returns the values for amount, currency and denomination (given the general capability to pay out the amount specified; see above).

   If, for example, the amount of £34.00 has been chosen, the CDM service will try to separate the required coins (up to the configured maximum value) using a coin dispenser. If no coin dispenser is available, the separated amount is assigned to sources other than the CDM (such as the teller's cash box). An ATM (having no cash box) raises an error indicating that this amount cannot be denominated.

   If, for example, there are no more £10.00 bills in the CDM (cash unit minimum has been reached), it performs payments using bills denominating £20.00 while the remaining £10.00 will have to be paid out from the cash box, if present.
4. Input parameters are currency, amount and mix number, where the desired denomination is partly defined or a minimum amount for the cashbox is specified. In these cases the partly specified denomination is completed; the cashbox amount may be updated and returned together with the amount and the desired currency.

The following errors can occur:

- If the denomination specified requires the selection of a locked cash unit, the CDM service returns the error WFS_ERR_CDM_CASHUNITERROR.
- If the sum total of the denomination is greater than the amount specified (exception: amount is zero), the CDM service returns the error WFS_ERR_CDM_INVALIDDENOMINATION.
- If the amount specified cannot be dispensed, either because a bill value or coin type is not in the machine, the difference is requested from the teller's cash box. If the device is an ATM, there is no cash box and the error WFS_ERR_CDM_NOTDISPENSABLE is returned.
- If no coin dispenser is in the unit and a coin amount is specified, a WFS_ERR_CDM_CASHUNITERROR error is returned.
- If the desired denomination refers to cash units containing different currencies, the CDM-service returns the error WFS_ERR_CDM_NOCURRENCYMIX. A cash unit with a currency type indicating a coupon or non-cash item can be mixed with other currencies.
- If the currency specified is not configured for the service, a WFS_ERR_CDM_INVALIDCURRENCY error is returned.

Input Param `LPWFSCDMDENOMINATE  lpDenominate;`

```
typedef struct _wfs_cdm_denominate
   {
   USHORT                  usTellerID;
   USHORT                  usMixNumber;
   LPWFSCDMDENOMINATION    lpDenomination;
   } WFSCDMDENOMINATE, * LPWFSCDMDENOMINATE;
```

usTellerID
Identification of teller.

usMixNumber
Mix algorithm or house mix table to be used. If the value is WFS_CDM_INDIVIDUAL, the service does not calculate an alternative denomination.

lpDenomination
Pointer to a WFSCDMDENOMINATION structure, describing the contents of the denomination operation.

```
typedef struct _wfs_cdm_denomination
   {
   CHAR          cCurrencyID[3];
   ULONG         ulAmount;
   USHORT        usCount;
   LPULONG       lpulValues;
   ULONG         ulCashBox;
   } WFSCDMDENOMINATION, * LPWFSCDMDENOMINATION;
```

cCurrencyID
Identification of currency (ISO format).

ulAmount
The total amount of money to be dispensed. (Amount expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP.)

usCount
Number of cash units used. Size of the array *lpulValues*.

lpulValues
Pointer to a list of ULONGs, specifying the number of coins/bills to take from the cash unit. The position in the list defines the logical number of the logical cash unit to be used, the first value in the array is related to the cash unit with the logical number 1. When more than one physical cash unit exists for a logical cash unit, the device selects the actual physical cash unit to use.

ulCashBox
Amount of money that could not be denominated and has to be paid from the tellers cash

box. (Amount expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP.)

Output Param `LPWFSCDMDENOMINATION lpDenomination;`
For a description see the input structure.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_CASHUNITERROR	The specified cash unit caused a problem. A WFS_EXECUTE_EVENT with an id of WFS_EXEE_CDM_CASHUNITERROR is posted with the details.
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE) .
WFS_ERR_CDM_INVALIDCURRENCY	Currency type is not configured.
WFS_ERR_CDM_INVALIDDENOMINATION	The sum of the values for cashbox, cash unit and coin were greater than the amount specified. Or, *usMixNumber* was WFS_CDM_INDIVIDUAL and the calculated denomination is smaller than the given amount.
WFS_ERR_CDM_INVALIDMIXNUMBER	Mix algorithm is not known.
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list
WFS_ERR_CDM_NOCURRENCYMIX	More than one currency was selected when the cash units were specified. The exception to this is when the cash unit selected contains a non-currency value such as a coupon.
WFS_ERR_CDM_NOTDISPENSABLE	The amount is not dispensable by the cash dispenser.
WFS_ERR_CDM_TOOMANYBILLS	The request would require too many bills to be dispensed.
WFS_ERR_CDM_TOOMANYCOINS	The request would require too many coins to be dispensed.

Events The following additional events can be generated by this command:

Value	Meaning
WFS_EXEE_CDM_CASHUNITERROR	An error occurred while attempting to denominate from the cash unit specified by the event.

Comments None.

4.2 WFS_CMD_CDM_DISPENSE

Description This command controls the dispensing of money. It requires specifications for the amount of the dispense (expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP), the desired denomination (or, alternatively, a procedure for the denomination) and the currency desired for the payout. If both the amount and the denomination have been specified, their consistency is checked, while a specification of amount, mix type and currency will produce a response that indicates the denomination. If the amount is not specified (amount is zero), but the denomination is, there is only a check for an approved denomination (as in WFS_CMD_CDM_DENOMINATE), then the dispense occurs.

Instead of using the input parameter *usPosition* (which is set to WFS_CDM_POSNULL in this case), the teller number can be used so that the teller list can be employed to perform the dispensing to the assigned teller position.

The WFS_CMD_CDM_DISPENSE command is essentially the same as the WFS_CMD_CDM_DENOMINATE command, the main difference between them being that, in addition to the denomination, the dispensing is performed, too. A configuration parameter (WFS_INF_CDM_CAPABILITIES cashbox) determines whether even if only part of the total amount is capable of being denominated, its dispensing will be performed by the CDM.

Examples:

1. If $30.00 is to be dispensed by the CDM but the smallest currency unit available is a $20 bill, it is possible to dispense $20.00 from the CDM, while the remaining $10.00 is requested from the teller's cash box.
2. The CDM service returns a message saying that the amount of a payout cannot be denominated (WFS_ERR_CDM_NOTDISPENSABLE).

Input Param

```
LPWFSCDMDISPENSE    lpDispense;

typedef struct _wfs_cdm_dispense
   {
   USHORT                  usTellerID;
   USHORT                  usMixNumber;
   USHORT                  usPosition;
   BOOL                    bPresent;
   LPWFSCDMDENOMINATION    lpDenomination;
   } WFSCDMDISPENSE, *LPWFSCDMDISPENSE;
```

usTellerID
Identification of teller.

usMixNumber
Mix algorithm or house mix table to be used. If the value is WFS_CDM_INDIVIDUAL, the service does not calculate an alternative denomination.

usPosition
Determines to which side the amount is dispensed; values are:

Value	Meaning
WFS_CDM_POSNULL	This implies that the default configuration information is used. This can be either position dependent or teller dependent for determining which side the currency is presented.
WFS_CDM_POSLEFT	Present money to left side of device.
WFS_CDM_POSRIGHT	Present money to right side of device.
WFS_CDM_POSCENTER	Present money to center output position.

bPresent
Controls whether the bills should be presented to the user (= TRUE) or only transported to the stacker (= FALSE). See WFS_CMD_CDM_PRESENT and WFS_CMD_CDM_REJECT.

lpDenomination
Pointer to a WFSCDMDENOMINATION structure, describing the denominations used for the dispense operation. For a description of the WFSCDMDENOMINATION structure see the definition of the command WFS_CMD_CDM_DENOMINATE.

Output Param

```
LPWFSCDMDENOMINATION        lpDenomination;
```

For a description of the WFSCDMDENOMINATION structure see the definition of the command WFS_CMD_CDM_DENOMINATE. Note that the values in this structure report the actual total amount and number of each denomination dispensed.

Error Codes

The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_CASHUNITERROR	A cash unit specified caused a problem, e.g., ran out of bills/coins (short dispense). A WFS_EXEE_CDM_CASHUNITERROR EXECUTE_EVENT is posted with the details.
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_INVALIDCURRENCY	Currency type is not configured

WFS_ERR_CDM_INVALIDDENOMINATION	The sum of the values for cashbox, cash unit and coin were greater than the amount specified.
WFS_ERR_CDM_INVALIDMIXNUMBER	Mix algorithm is not known
WFS_ERR_CDM_INVALIDPOSITION	The specified output position is invalid.
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list
WFS_ERR_CDM_NOCURRENCYMIX	Cash units containing two or more different currencies were selected. This error is not generated if one or more non-currency value cash units (e.g., containing coupons) are selected, together with currency cash units of a single currency type.
WFS_ERR_CDM_NOTDISPENSABLE	The requested amount is not dispensable by the cash dispenser.
WFS_ERR_CDM_POSITIONLOCKED	The output position is locked.
WFS_ERR_CDM_SAFEDOOROPEN	The safe door is open.
WFS_ERR_CDM_TOOMANYBILLS	The request would require too many bills to be dispensed.
WFS_ERR_CDM_TOOMANYCOINS	The request would require too many coins to be dispensed.

Events The following additional events can be generated:

Value	Meaning
WFS_EXEE_CDM_DELAYEDDISPENSE	Time before dispensing starts, because of a security procedure (German national regulation).
WFS_EXEE_CDM_STARTDISPENSE	Point of time where the dispense order starts. Necessary to know for a CDM application because of queueing orders from different clients.
WFS_EXEE_CDM_PARTIALDISPENSE	The dispense order is divided into several suborders.
WFS_EXEE_CDM_SUBDISPENSEOK	One of the dispense suborders was finished successfully.
WFS_EXEE_CDM_CASHUNITERROR	An error occurred while attempting to dispense cash.

Comments All error codes and events listed under the WFS_CMD_CDM_PRESENT command description, with the exception of WFS_ERR_CDM_NOBILLS, can also occur on this command**.**

4.3 WFS_CMD_CDM_PRESENT

Description This command is only used for ATMs; it causes presentation of the currency. It can be used only following the WFS_CMD_CDM_DISPENSE command (with *bPresent* = FALSE).

The command completes when the bills are positioned at the exit slot of the device. A service event is generated to report the user has removed the bills. If no event is received within a reasonable time period, the application should send a WFS_CMD_CDM_RETRACT to clear the bills from the exit. On devices which do not have the ability to detect when bills are taken (see WFS_INF_CDM_CAPABILITIES) the service event is generated as soon as the bills are available to the user.

Input Param None.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_NOBILLS	There were no bills on the stacker to present.
WFS_ERR_CDM_SHUTTERNOTOPEN	The shutter is not open or did not open when it should have. No bills presented.

WFS_ERR_CDM_SHUTTEROPEN	The shutter is open when it should be closed. No bills presented.
WFS_ERR_CDM_PRERRORNOBILLS	There was an error during the present operation - no bills are presented.
WFS_ERR_CDM_PRERRORBILLS	There was an error during the present operation - at least some of the bills are presented.
WFS_ERR_CDM_PRERRORUNKNOWN	There was an error during the present operation - the position of the bills is unknown. Intervention may be required to reconcile the cash amount totals.

Events The following additional events can be generated:

Value	Meaning
WFS_SRVE_CDM_BILLSTAKEN	The bills presented have been removed by the user.

Comments None**.**

4.4 WFS_CMD_CDM_REJECT

Description This command is used only in ATMs. It causes money to be transported from the stacker into the reject bin. It can be used only following the WFS_CMD_CDM_DISPENSE command (with *bPresent* = FALSE).

Input Param None.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_NOBILLS	There were no bills on the stacker to reject.

Events There are no additional events generated by this command**.**

Comments None**.**

4.5 WFS_CMD_CDM_RETRACT

Description This command is used only for ATMs. It allows the application to force cash that has been presented to be retracted. Not all cash dispensers support this capability.

Input Param `LPUSHORT lpusRetractArea;`
Pointer to value indicating the area in the reject bin where retracted cash is to be stored. Not all systems require this parameter, and it may differ in content from system to system (NULL if not relevant).

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_BILLSTAKEN	Bills were present at the exit at the start of the operation, but were removed before the operation was complete, so some or all of the bills were not retracted.
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_INVALIDRETRACT	Retract function is invalid for this system.
WFS_ERR_CDM_SHUTTERNOTCLOSED	The shutter failed to close.
WFS_ERR_CDM_NOBILLS	There were no presented bills to retract.

Events There are no additional events generated by this command**.**

Comments None**.**

4.6 WFS_CMD_CDM_CASH_IN

Description This command is not used for ATMs; there are two possibilities for use of this API.

The amount to be deposited is expressed in minimum dispense units; see WFS_INF_CDM_CURRENCY_EXP. It affects the teller counters only. The input parameters *cCurrencyID* and *ulAmount* are required as input.

If the hardware is capable of identifying the currency and the denomination of the amount deposited, the output parameters *cCurrencyID*, *ulAmount*, *lpulValues*, and *ulCashBox* will be returned.

If the hardware is equipped with refill containers the amount cashed in is simply placed on top of the cash already in the refill containers. The input parameters *cCurrencyID*, *ulAmount* and *lpulValues* are required.

Input Param

```
LPWFSCDMCASHIN      lpCashIn;

   typedef struct _wfs_cdm_cashin
      {
      USHORT                  usTellerID;
      LPWFSCDMDENOMINATION    lpDenomination;
      } WFSCDMCASHIN, * LPWFSCDMCASHIN;
```

usTellerID
Identification of Teller.

lpDenomination
Pointer to a WFSCDMDENOMINATION structure, describing the denomination of the cash in operation. For a description of the WFSCDMDENOMINATION structure see the definition of the WFS_CMD_CDM_DENOMINATE function.

Output Param

```
LPWFSCDMDENOMINATION       lpDenomination;
```

For a description of the WFSCDMDENOMINATION structure see the definition of the command WFS_CMD_CDM_DENOMINATE.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_INVALIDCURRENCY	Specified currency not currently available
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list
WFS_ERR_CDM_NOCASHINSTARTED	The WFS_CMD_CDM_CASH_IN_START was not issued before

Events The following additional events can be generated:

Value	Meaning
WFS_EXEE_CDM_INPUTREFUSE	A part of the amount of the cash in order was refused.

Comments None.

4.7 WFS_CMD_CDM_OPEN_SHUTTER

Description This command is used only for ATMs. It opens the shutter.

Input Param None.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_SHUTTERNOTOPEN	Shutter failed to open
WFS_ERR_CDM_SHUTTEROPEN	Shutter was already open

Events There are no additional events generated by this command**.**

Comments None**.**

4.8 WFS_CMD_CDM_CLOSE_SHUTTER

Description This command is used only for ATMs. It closes the shutter.

Input Param None.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_SHUTTERCLOSED	Shutter was already closed
WFS_ERR_CDM_SHUTTERNOTCLOSED	Shutter failed to close

Events There are no additional events generated by this command**.**

Comments None**.**

4.9 WFS_CMD_CDM_SET_TELLER_INFO

Description This command is used for initializing the tallies assigned to a teller. All values are absolute. For each currency a different set of tallies is used.

Input Param
```
LPWFSCDMTELLERTOTALS        lpTellerTotals;
```
For a description of the struct WFSCDMTELLERTOTALS see the definition of the WFS_INF_CDM_TELLER_INFO command.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_INVALIDCURRENCY	Specified currency not currently available
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list

Events The following additional events can be generated by this command:

Value	Meaning
WFS_SRVE_CDM_TELLERINFOCHANGED	Teller information has been changed.

Comments None**.**

4.10 WFS_CMD_CDM_SET_CASH_UNIT_INFO

Description This command is used to adjust both cash unit counters and cash unit IDs. In addition to that, application locks for cash units can be either installed or removed. It is also useful for setting or altering threshold values of cassettes.

This command can be used when a problem has occurred and the start and end cash unit exchange is not acceptable.

The following parameters defined in the WFSCDMCASHUNIT structure cannot be set by the application and, therefore, are ignored by this command:
bDevLock
usStatus
lpPhysicalPositionName
lppPhysical[]->usPStatus
lppPhysical[]->lpPhysicalPositionName

Input Param `LPWFSCDMCUINFO lpCUInfo;`
For a description of the WFSCDMCUINFO structure, see the WFS_INF_CDM_CASH_UNIT_INFO command.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_CASHUNITERROR	A cash unit specified caused a problem. A WFS_EXEE_CDM_CASHUNITERROR execute event is posted with the details.
WFS_ERR_CDM_INVALIDCASHUNIT	Invalid cash unit ID.
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).

Events The following additional events can be generated:

Value	Meaning
WFS_SRVE_CDM_CASHUNITINFOCHANGED	A cash unit was changed.

Comments This command generates a service event (WFS_SRVE_CDM_CASHUNITINFOCHANGED) that allows applications to be aware that they should update their cash unit status for the service

4.11 WFS_CMD_CDM_START_EXCHANGE

Description This command is used to start the exchange of cash units as well as their emptying, replenishment, removal or replacement. The command returns the current values in the device and the device itself initiates cash unit removal (for example by means of lowering the cash units). A **WFSLock** should be performed before this command is initiated, to ensure exclusive control by the replenishment application.

After WFS_CMD_CDM_START_EXCHANGE has been successfully initiated, the CDM enters the exchange state, and accepts only the following functions:

- **WFS[Async]Execute**, with WFS_CMD_CDM_END_EXCHANGE command only
- **WFS[Async]GetInfo** commands
- **WFSClose**

Any other commands issued when the CDM service is in the exchange state are rejected with an error condition of WFS_ERR_CDM_EXCHANGEACTIVE. If an error occurs at the CDM during the execution of this command, the cash unit values are not returned.

Input Param `LPWFSCDMSTARTEX lpStartEx;`

```
typedef struct _wfs_cdm_start_ex
   {
   USHORT       usTellerID;
   USHORT       usCount;
   LPUSHORT     lpusCUNumList;
   } WFSCDMSTARTEX, * LPWFSCDMSTARTEX;
```

usTellerID
Identification of teller.

usCount
Number of cash units to be exchanged. Size of the array *lpusCUNumList*.

lpusCUNumList
Pointer to a list of the logical numbers (unsigned shorts) of the cash units to be exchanged.

Output Param `LPWFSCDMCUINFO lpCUInfo;`
For a description of the WFSCDMCUINFO structure, see the definition of the WFS_INF_CDM_CASH_UNIT_INFO command.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is already in an exchange state.
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list

Events There are no additional events generated by this command**.**

Comments None**.**

4.12 WFS_CMD_CDM_END_EXCHANGE

Description This command is initiated after a cash unit has been exchanged, refilled, etc. It supplies the latest cash unit information and cash unit IDs. Cash units are set as ready (e.g., lifted upwards; there may also be a dispenser test), any cash unit inoperative conditions are cleared, and reject counters are reset; the current status of the safe door is ignored. After this command the cash dispenser service will accept a **WFS[Async]Unlock** request.

The service provider may be able to obtain cash unit information directly from the dispenser device, such as cash unit ID and replenishment state. In this case, the corresponding input parameters to this command may be ignored.

Errors can be generated by this command. When a cash unit error is returned, the application must issue a WFS_INF_CDM_CASH_UNIT_INFO command to get the cash unit status.

Input Param `LPWFSCDMCUINFO lpCUInfo;`
For a description of the WFSCDMCUINFO structure, see the definition of the WFS_INF_CDM_CASH_UNIT_INFO command. If this pointer is NULL, the contents of the cash units are not changed.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_NOEXCHANGEACTIVE	There is no exchange active
WFS_ERR_CDM_INVALIDCASHUNIT	Invalid cash unit ID
WFS_ERR_CDM_INVALIDTELLERID	Teller ID not present in service provider's teller ID list

Events The following additional events can be generated by this command:

Value	Meaning
WFS_SRVE_CDM_CASHUNITINFOCHANGED	A cash unit was changed.

Comments This command generates a service event (WFS_SRVE_CDM_CASHUNITINFOCHANGED) that allows applications to be aware that they should update their cash unit status for the service, using the WFS_INF_CDM_CASH_UNIT_INFO command.

4.13 WFS_CMD_CDM_OPEN_SAFE_DOOR

Description This command controls the time lock for the safe door. It sends the currently configured value for the safe door timer to the device. This configuration parameter is vendor dependent.

Input Param None.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).

Events There are no additional events generated by this command**.**

Comments None**.**

4.14 WFS_CMD_CDM_CHECK_VANDALISM

Description This command is used only for ATMs. If implemented, it checks whether there has been any attempt to manipulate the ATM. If vandalism was detected, it is reported through this function.

Input Param None.

Output Param `LPUSHORT lpusVandalism;`

Flag specifying whether there has been an attempt to manipulate the device since the last time this command was issued. Values are:

Value	Meaning
WFS_CDM_NODEVMANIPULATION	No attempt has been made to manipulate the device.
WFS_CDM_DEVMANIPULATION	An attempt to manipulate the device has been detected.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).

Events There are no additional events generated by this command**.**

Comments None**.**

4.15 WFS_CMD_CDM_CALIBRATE_CASH_UNIT

Description This command is used to initialize the reference value of a cash unit. It will action a vendor dependent sequence of hardware events which will calibrate the physical cash unit. This is necessary if a new type of bank note is put into the cash unit. By this command the cash unit gets the new measures of the bank notes.

If more than one physical cash unit is related to the logical cash unit, it is up to the Service Provider to select the appropriate actions. Whether, all the physical cash units have to be calibrated or if it is sufficient to do the calibration for one physical unit and load the data into the others.

Input Param `LPWFSCDMCALIBRATE lpCalibrateIn;`

```
typedef struct _wfs_cdm_calibrate
  {
  USHORT       usNumber;
  USHORT       usNumOfBills;
  } WFSCDMCALIBRATE, * LPWFSCDMCALIBRATE;
```

usNumber
Logical number of cash unit.

usNumOfBills
Number of bills to be dispensed during the process of measuring.

Output Param `LPWFSCDMCALIBRATE lpCalibrateOut;`

For a description of the WFSCDMCALIBRATE structure, see the definition of the Input Param.

usNumber
Logical number of cash unit.

usNumOfBills
Number of bills that were dispensed during the process of measuring. It can differ from the value in the input structure, if the cash dispenser always dispenses a default number of bills.

Error Codes The following additional error code can be generated by this command:

Value	Meaning
WFS_ERR_CDM_EXCHANGEACTIVE	The CDM service is in an exchange state (see WFS_CMD_CDM_START_EXCHANGE).
WFS_ERR_CDM_CASHUNITERROR	The specified cash unit caused an error.

Events The following additional events can be generated by this command:

Value	Meaning
WFS_SRVE_CDM_CASHUNITINFOCHANGED	A cash unit was changed.

Comments None**.**

4.16 WFS_CMD_CDM_SET_TELLER_POSITIONS

Description This command is used to set the output position assigned to each teller.

Input Param
```
LPWFSCDMTELLERPOS    *lppTellerPos;
```
For a description of the struct WFSCDMTELLERPOS see the definition of the command WFS_INF_CDM_TELLER_POSITIONS.

Output Param None.

Error Codes There are no additional error codes generated by this command.

Events There are no additional events generated by this command**.**

Comments None**.**

4.17 WFS_CMD_CDM_CASH_IN_START

Description Each cash in procedure has to be handled as a transaction that can be rolled back if a difference occurs between the amount counted by the ATS and the amount the teller inserted. This command is used to start the cash in transaction.

Input Param
```
LPUSHORT      lpusTellerID;
```
Identification of teller.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_CASHINACTIVE	The CDM service has already got a WFS_CMD_CDM_CASH_IN_START command.

Events There are no additional events generated by this command**.**

Comments None**.**

4.18 WFS_CMD_CDM_CASH_IN_END

Description Each cash in procedure has to be handled as a transaction that can be rolled back if a difference occurs between the amount counted by the ATS and the amount the teller inserted. This command is used to end the cash in transaction.

Input Param `LPUSHORT lpusTellerID;`
Identification of teller. It has to be the same as issued in the WFS_CMD_CDM_CASH_IN_START command.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_NOCASHINSTARTED	The WFS_CMD_CDM_CASH_IN_START was not issued before

Events There are no additional events generated by this command**.**

Comments None**.**

4.19 WFS_CMD_CDM_CASH_IN_ROLLBACK

Description Each cash in procedure has to be handled as a transaction that can be rolled back if a difference occurs between the amount counted by the ATS and the amount the teller inserted. This command is used to roll back the cash in transaction. All the notes cashed in since the last WFS_CMD_CDM_CASH_IN_START command are returned to the teller. If an ATS does not have this capability, it returns 0 number of bills in the output structure.

Input Param `LPUSHORT lpusTellerID;`
Identification of teller. It has to be the same as issued in the WFS_CMD_CDM_CASH_IN_START command.

Output Param LPWFSCDMDENOMINATION lpDenomination;
For a description of the struct WFSCDMDENOMINATION see the definition of the command WFS_CMD_CDM_DENOMINATE.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_NOCASHINSTARTED	The WFS_CMD_CDM_CASH_IN_START was not issued before

Events There are no additional events generated by this command**.**

Comments None**.**

4.20 WFS_CMD_CDM_SET_MIX_TABLE

Description This command is used to set up the mix table specified by the mix number.

Input Param `LPWFSCDMMIXTABLE lpMixTable;`
For a description of the struct WFSCDMMIXTABLE see the definition of the command WFS_INF_CDM_MIX_TABLE.

Output Param None.

Error Codes The following additional error codes can be generated by this command:

Value	Meaning
WFS_ERR_CDM_INVALIDMIXNUMBER	The *usMixNumber* parameter refers to a predefined and reserved mix algorithm.
WFS_ERR_CDM_INVALIDMIXTABLE	The contents of at least one of the defined rows of the mix table is incorrect.

Events There are no additional events generated by this command**.**

Comments None**.**

5. Events

5.1 WFS_SRVE_CDM_SAFEDOOROPEN

Description This service event specifies that the safe door has been opened.

Event Param None.

Comments None**.**

5.2 WFS_SRVE_CDM_SAFEDOORCLOSED

Description This service event specifies that the safe door has been closed.

Event Param None.

Comments None**.**

5.3 WFS_USRE_CDM_CASHUNITTHRESHOLD

Description This user event specifies that a threshold condition has occured in one of the cash units.

Event Param
```
LPWFSCDMCASHUNIT    lpCashUnit;
```

lpCashUnit
Pointer to WFSCDMCASHUNIT structure, describing the cash unit on which the threshold condition occurred. See *lpCashUnit->usStatus* for the type of condition. For a description of the WFSCDMCASHUNIT structure, see the definition of the WFS_INF_CDM_CASH_UNIT_INFO command.

Comments None**.**

5.4 WFS_SRVE_CDM_CASHUNITINFOCHANGED

Description This service event specifies that a cash unit was exchanged, a new cash unit was added, or the contents of a cash unit were modified using the WFS_CMD_CDM_SET_CASH_UNIT_INFO command. This event is also posted after a WFS_CMD_CDM_END_EXCHANGE or a WFS_CMD_CDM_CALIBRATE_CASH_UNIT command successfully completes.

Event Param
```
LPWFSCDMCASHUNIT    lpCashUnit;
```

lpCashUnit
Pointer to the changed cash unit structure. For a description of the WFSCDMCASHUNIT structure see the definition of the WFS_INF_CDM_CASH_UNIT_INFO command.

Comments None**.**

5.5 WFS_SRVE_CDM_TELLERINFOCHANGED

Description This service event specifies that the tallies assigned to the specified teller have been changed. This event is only returned as a result of a WFS_CMD_CDM_SET_TELLER_INFO command.

Event Param
```
LPUSHORT    lpusTellerID;
```

lpusTellerID
Pointer to an unsigned short holding the ID of the teller whose tallies have been changed.

Comments None**.**

5.6 WFS_EXEE_CDM_DELAYEDDISPENSE

Description This execute event specifies that the start of the physical dispensing of the money has been delayed.

Event Param
```
LPULONG       lpulDelay;
```

lpulDelay
Pointer to the time in milliseconds the dispense job will be delayed.

Comments None**.**

5.7 WFS_EXEE_CDM_STARTDISPENSE

Description This execute event specifies the start of the physical dispensing of the money of the formerly delayed job.

Event Param
```
LPREQUESTID   lpReqID;
```

lpReqID
Pointer to the *RequestID* of the dispense command that has started.

Comments None**.**

5.8 WFS_EXEE_CDM_CASHUNITERROR

Description This execute event specifies that in a denominate or dispense command a cash unit was addressed which caused a problem.

Event Param
```
LPWFSCDMCUERROR     lpCashUnitError;

typedef struct _wfs_cdm_cu_error
   {
   WORD                wFailure;
   LPWFSCDMCASHUNIT    lpCashUnit;
   } WFSCDMCUERROR, * LPWFSCDMCUERROR;
```

wFailure
Specifies the kind of failure that occurred in the cash unit. Values are:

Value	Meaning
WFS_CDM_CASHUNITEMPTY	Specified cash unit is empty.
WFS_CDM_CASHUNITERROR	Specified cash unit has malfunctioned.
WFS_CDM_CASHUNITFULL	Specified cash unit is full.
WFS_CDM_CASHUNITLOCKED	Specified cash unit is locked.
WFS_CDM_CASHUNITNOTCONF	Specified cash unit is not configured.
WFS_CDM_CASHUNITINVALID	Specified cash unit ID is invalid.

lpCashUnit
Pointer to the cash unit structure that caused the problem. For a description of the WFSCDMCASHUNIT structure see the definition of the WFS_INF_CDM_CASH_UNIT_INFO command.

Comments None.

5.9 WFS_SRVE_CDM_BILLSTAKEN

Description This service event specifies that the bills presented to the user have been taken.

Event Param None.

Comments None**.**

5.10 WFS_EXEE_CDM_PARTIALDISPENSE

Description This execute event specifies that the dispense order is divided into several suborders because the capacity of the device is exceeded.

Event Param `LPUSHORT lpusDispNum;`

lpusDispNum
Specifies the number of suborders into which the dispense order is divided.

Comments None**.**

5.11 WFS_EXEE_CDM_SUBDISPENSEOK

Description This execute event specifies that one of the suborders into which the dispense order was divided was finished successfully.

Event Param `LPWFSCDMDENOMINATION lpDenomination;`

lpDenomination
For a description of the struct WFSCDMDENOMINATION see the definition of the command WFS_CMD_CDM_DENOMINATE. Note that in this case the values in this structure report the amount and number of each denomination dispensed in the suborder this event belongs to.

Comments None**.**

5.12 WFS_EXEE_CDM_INPUTREFUSE

Description This execute event specifies that the device has refused a part of the amount of the cash in order.

Event Param `LPUSHORT lpusReason;`

lpusReason
Specifies the reason for refusing a part of the amount. Possible values are:.

Value	Meaning
WFS_CDM_CASHUNITFULL	Cash unit is full.
WFS_CDM_INVALIDBILL	One or more of the bills are invalid.

Comments None**.**

6. C - Header file

```
/******************************************************************************
*                                                                             *
* xfscdm.h      XFS - Cash Dispenser (CDM) definitions                        *
*                                                                             *
*               Version 2.00 (11/11/96)                                       *
*                                                                             *
******************************************************************************/

#ifndef __INC_XFSCDM__H
#define __INC_XFSCDM__H

#ifdef __cplusplus
extern "C" {
#endif

#include <xfsapi.h>

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

/* values of WFSCDMCAPS.wClass */

#define     WFS_SERVICE_CLASS_CDM               (3)
#define     WFS_SERVICE_CLASS_VERSION_CDM       0x0002
#define     WFS_SERVICE_CLASS_NAME_CDM          "CDM"

#define     CDM_SERVICE_OFFSET                  (WFS_SERVICE_CLASS_CDM * 100)

/* CDM Info Commands */

#define     WFS_INF_CDM_STATUS                  (CDM_SERVICE_OFFSET + 1)
#define     WFS_INF_CDM_CAPABILITIES            (CDM_SERVICE_OFFSET + 2)
#define     WFS_INF_CDM_CASH_UNIT_INFO          (CDM_SERVICE_OFFSET + 3)
#define     WFS_INF_CDM_TELLER_INFO             (CDM_SERVICE_OFFSET + 4)
#define     WFS_INF_CDM_TELLER_POSITIONS        (CDM_SERVICE_OFFSET + 5)
#define     WFS_INF_CDM_CURRENCY_EXP            (CDM_SERVICE_OFFSET + 6)
#define     WFS_INF_CDM_MIX_TYPES               (CDM_SERVICE_OFFSET + 7)
#define     WFS_INF_CDM_MIX_TABLE               (CDM_SERVICE_OFFSET + 8)
#define     WFS_INF_CDM_PRESENT_STATUS          (CDM_SERVICE_OFFSET + 9)

/* CDM Execute Commands */

#define     WFS_CMD_CDM_DENOMINATE              (CDM_SERVICE_OFFSET + 1)
#define     WFS_CMD_CDM_DISPENSE                (CDM_SERVICE_OFFSET + 2)
#define     WFS_CMD_CDM_PRESENT                 (CDM_SERVICE_OFFSET + 3)
#define     WFS_CMD_CDM_REJECT                  (CDM_SERVICE_OFFSET + 4)
#define     WFS_CMD_CDM_RETRACT                 (CDM_SERVICE_OFFSET + 5)
#define     WFS_CMD_CDM_CASH_IN                 (CDM_SERVICE_OFFSET + 6)
#define     WFS_CMD_CDM_OPEN_SHUTTER            (CDM_SERVICE_OFFSET + 7)
#define     WFS_CMD_CDM_CLOSE_SHUTTER           (CDM_SERVICE_OFFSET + 8)
#define     WFS_CMD_CDM_SET_TELLER_INFO         (CDM_SERVICE_OFFSET + 9)
#define     WFS_CMD_CDM_SET_CASH_UNIT_INFO      (CDM_SERVICE_OFFSET + 10)
#define     WFS_CMD_CDM_START_EXCHANGE          (CDM_SERVICE_OFFSET + 11)
#define     WFS_CMD_CDM_END_EXCHANGE            (CDM_SERVICE_OFFSET + 12)
#define     WFS_CMD_CDM_OPEN_SAFE_DOOR          (CDM_SERVICE_OFFSET + 13)
#define     WFS_CMD_CDM_CHECK_VANDALISM         (CDM_SERVICE_OFFSET + 14)
#define     WFS_CMD_CDM_CALIBRATE_CASH_UNIT     (CDM_SERVICE_OFFSET + 15)
#define     WFS_CMD_CDM_SET_TELLER_POSITIONS    (CDM_SERVICE_OFFSET + 16)
#define     WFS_CMD_CDM_CASH_IN_START           (CDM_SERVICE_OFFSET + 17)
#define     WFS_CMD_CDM_CASH_IN_END             (CDM_SERVICE_OFFSET + 18)
#define     WFS_CMD_CDM_CASH_IN_ROLLBACK        (CDM_SERVICE_OFFSET + 19)
#define     WFS_CMD_CDM_SET_MIXTABLE            (CDM_SERVICE_OFFSET + 20)

/* CDM Messages */

#define     WFS_SRVE_CDM_SAFEDOOROPEN           (CDM_SERVICE_OFFSET + 1)
#define     WFS_SRVE_CDM_SAFEDOORCLOSED         (CDM_SERVICE_OFFSET + 2)
#define     WFS_USRE_CDM_CASHUNITTHRESHOLD      (CDM_SERVICE_OFFSET + 3)
#define     WFS_SRVE_CDM_CASHUNITINFOCHANGED    (CDM_SERVICE_OFFSET + 4)
#define     WFS_SRVE_CDM_TELLERINFOCHANGED      (CDM_SERVICE_OFFSET + 5)
#define     WFS_EXEE_CDM_DELAYEDDISPENSE        (CDM_SERVICE_OFFSET + 6)
```

```
#define     WFS_EXEE_CDM_STARTDISPENSE          (CDM_SERVICE_OFFSET + 7)
#define     WFS_EXEE_CDM_CASHUNITERROR          (CDM_SERVICE_OFFSET + 8)
#define     WFS_SRVE_CDM_BILLSTAKEN             (CDM_SERVICE_OFFSET + 9)
#define     WFS_EXEE_CDM_PARTIALDISPENSE        (CDM_SERVICE_OFFSET + 10)
#define     WFS_EXEE_CDM_SUBDISPENSEOK          (CDM_SERVICE_OFFSET + 11)
#define     WFS_EXEE_CDM_INPUTREFUSE            (CDM_SERVICE_OFFSET + 12)

/* values of WFSCDMSTATUS.fwDevice */
#define     WFS_CDM_DEVONLINE                   WFS_STAT_DEVONLINE
#define     WFS_CDM_DEVOFFLINE                  WFS_STAT_DEVOFFLINE
#define     WFS_CDM_DEVPOWEROFF                 WFS_STAT_DEVPOWEROFF
#define     WFS_CDM_DEVBUSY                     WFS_STAT_DEVBUSY
#define     WFS_CDM_DEVNODEVICE                 WFS_STAT_DEVNODEVICE
#define     WFS_CDM_DEVHWERROR                  WFS_STAT_DEVHWERROR
#define     WFS_CDM_DEVUSERERROR                WFS_STAT_DEVUSERERROR

/* values of WFSCDMSTATUS.fwSafeDoor */

#define     WFS_CDM_DOORNOTSUPPORTED            (1)
#define     WFS_CDM_DOOROPEN                    (2)
#define     WFS_CDM_DOORCLOSED                  (3)
#define     WFS_CDM_DOORLOCKED                  (4)
#define     WFS_CDM_DOORUNKNOWN                 (5)

/* values of WFSCDMSTATUS.fwDispenser */

#define     WFS_CDM_DISPOK                      (0)
#define     WFS_CDM_DISPCUSTATE                 (1)
#define     WFS_CDM_DISPCUSTOP                  (2)
#define     WFS_CDM_DISPCUUNKNOWN               (3)

/* values of WFSCDMSTATUS.fwShutter */

#define     WFS_CDM_SHTCLOSED                   (0)
#define     WFS_CDM_SHTOPEN                     (1)
#define     WFS_CDM_SHTJAMMED                   (2)
#define     WFS_CDM_SHTUNKNOWN                  (3)
#define     WFS_CDM_SHTNOTSUPPORTED             (4)

/* values of WFSCDMSTATUS.fwOutputPosition */

#define     WFS_CDM_CTEMPTY                     (0)
#define     WFS_CDM_CTNOTEMPTY                  (1)
#define     WFS_CDM_CTUNKNOWN                   (2)
#define     WFS_CDM_CTNOTSUPPORTED              (3)

/* values of WFSCDMSTATUS.fwTransport */

#define     WFS_CDM_TPOK                        (0)
#define     WFS_CDM_TPINOP                      (1)
#define     WFS_CDM_TPUNKNOWN                   (2)
#define     WFS_CDM_TPNOTSUPPORTED              (3)

/* values of WFSCDMSTATUS.fwIntermediateStacker */

#define     WFS_CDM_ISEMPTY                     (0)
#define     WFS_CDM_ISNOTEMPTY                  (1)
#define     WFS_CDM_ISUNKNOWN                   (2)
#define     WFS_CDM_ISNOTSUPPORTED              (3)

/* values of WFSCDMCAPS.fwType */

#define     WFS_CDM_TYPEATSAFE                  (1)
#define     WFS_CDM_TYPEATMACHINE               (2)

/* values of WFSCDMCASHUNIT.usType */

#define     WFS_CDM_TYPENA                      (1)
#define     WFS_CDM_TYPEREJECTCASSETTE          (2)
#define     WFS_CDM_TYPEBILLCASSETTE            (3)
#define     WFS_CDM_TYPECOINCYLINDER            (4)
#define     WFS_CDM_TYPECOINDISPENSER           (5)
#define     WFS_CDM_TYPERETRACTCASSETTE         (6)
#define     WFS_CDM_TYPECOUPON                  (7)
```

```
#define     WFS_CDM_TYPEDOCUMENT                (8)

/* values of WFSCDMCASHUNIT.usStatus */

#define     WFS_CDM_STATCUOK                    (0)
#define     WFS_CDM_STATCUFULL                  (1)
#define     WFS_CDM_STATCUHIGH                  (2)
#define     WFS_CDM_STATCULOW                   (3)
#define     WFS_CDM_STATCUEMPTY                 (4)
#define     WFS_CDM_STATCUMISSING               (5)
#define     WFS_CDM_STATCUINOP                  (6)
#define     WFS_CDM_STATCUNOVAL                 (7)
#define     WFS_CDM_STATCUNOREF                 (8)

/* values of WFSCDMMIXTYPE.usMixType */

#define     WFS_CDM_MIXALGORITHM                (1)
#define     WFS_CDM_MIXTABLE                    (2)

/* values of WFSCDMMIXTYPE.usMixNumber */

#define     WFS_CDM_INDIVIDUAL                  (0)

/* values of WFSCDMMIXTYPE.usSubType (predefined mix algorithms) */
#define     WFS_CDM_MIX_MINIMUM_NUMBER_OF_BILLS           (1)
#define     WFS_CDM_MIX_EQUAL_EMPTYING_OF_CASH_UNITS      (2)

/* values of WFSCDMPRESENTSTATUS.wPresentState */

#define     WFS_CDM_PRESENTED                   (1)
#define     WFS_CDM_NOTPRESENTED                (2)
#define     WFS_CDM_UNKNOWN                     (3)

/* values of WFSCDMDISPENSE.usPosition */
/* values of WFSCDMCAPS.fwOutputPositions */
/* values of WFSCDMOUTPOS.fwPosition */
/* values of WFSCDMTELLERPOS.usPosition */

#define     WFS_CDM_POSNULL                     (0x0000)
#define     WFS_CDM_POSLEFT                     (0x0001)
#define     WFS_CDM_POSRIGHT                    (0x0002)
#define     WFS_CDM_POSCENTER                   (0x0004)

/* values of lpusVandalism */

#define     WFS_CDM_NODEVMANIPULATION           (0)
#define     WFS_CDM_DEVMANIPULATION             (1)

/* values of WFSCDMCUERROR.wFailure */

#define     WFS_CDM_CASHUNITEMPTY               (1)
#define     WFS_CDM_CASHUNITLOCKED              (2)
#define     WFS_CDM_CASHUNITNOTCONF             (3)
#define     WFS_CDM_CASHUNITINVALID             (4)
#define     WFS_CDM_CASHUNITERROR               (5)
#define     WFS_CDM_CASHUNITFULL                (6)

/* values of lpusReason in WFS_EXEE_CDM_INPUTREFUSE */
#define     WFS_CDM_INVALIDBILL                 (7)

/* XFS CDM Errors */

#define WFS_ERR_CDM_INVALIDCURRENCY     (-(CDM_SERVICE_OFFSET + 0))
#define WFS_ERR_CDM_INVALIDTELLERID     (-(CDM_SERVICE_OFFSET + 1))
#define WFS_ERR_CDM_CASHUNITERROR       (-(CDM_SERVICE_OFFSET + 2))
#define WFS_ERR_CDM_INVALIDDENOMINATION (-(CDM_SERVICE_OFFSET + 3))
#define WFS_ERR_CDM_INVALIDMIXNUMBER    (-(CDM_SERVICE_OFFSET + 4))
#define WFS_ERR_CDM_NOCURRENCYMIX       (-(CDM_SERVICE_OFFSET + 5))
#define WFS_ERR_CDM_NOTDISPENSABLE      (-(CDM_SERVICE_OFFSET + 6))
#define WFS_ERR_CDM_TOOMANYBILLS        (-(CDM_SERVICE_OFFSET + 7))
#define WFS_ERR_CDM_INVALIDPOSITION     (-(CDM_SERVICE_OFFSET + 8))
#define WFS_ERR_CDM_POSITIONLOCKED      (-(CDM_SERVICE_OFFSET + 9))
#define WFS_ERR_CDM_SAFEDOOROPEN        (-(CDM_SERVICE_OFFSET + 10))
#define WFS_ERR_CDM_INVALIDRETRACT      (-(CDM_SERVICE_OFFSET + 11))
```

```
#define WFS_ERR_CDM_SHUTTERNOTOPEN      (-(CDM_SERVICE_OFFSET + 12))
#define WFS_ERR_CDM_SHUTTEROPEN         (-(CDM_SERVICE_OFFSET + 13))
#define WFS_ERR_CDM_SHUTTERCLOSED       (-(CDM_SERVICE_OFFSET + 14))
#define WFS_ERR_CDM_INVALIDCASHUNIT     (-(CDM_SERVICE_OFFSET + 15))
#define WFS_ERR_CDM_NOBILLS             (-(CDM_SERVICE_OFFSET + 16))
#define WFS_ERR_CDM_EXCHANGEACTIVE      (-(CDM_SERVICE_OFFSET + 17))
#define WFS_ERR_CDM_NOEXCHANGEACTIVE    (-(CDM_SERVICE_OFFSET + 18))
#define WFS_ERR_CDM_SHUTTERNOTCLOSED    (-(CDM_SERVICE_OFFSET + 19))
#define WFS_ERR_CDM_PRERRORNOBILLS      (-(CDM_SERVICE_OFFSET + 20))
#define WFS_ERR_CDM_PRERRORBILLS        (-(CDM_SERVICE_OFFSET + 21))
#define WFS_ERR_CDM_PRERRORUNKNOWN      (-(CDM_SERVICE_OFFSET + 22))
#define WFS_ERR_CDM_BILLSTAKEN          (-(CDM_SERVICE_OFFSET + 23))
#define WFS_ERR_CDM_TOOMANYCOINS        (-(CDM_SERVICE_OFFSET + 24))
#define WFS_ERR_CDM_CASHINACTIVE        (-(CDM_SERVICE_OFFSET + 25))
#define WFS_ERR_CDM_NOCASHINSTARTED     (-(CDM_SERVICE_OFFSET + 26))
#define WFS_ERR_CDM_INVALIDMIXTABLE     (-(CDM_SERVICE_OFFSET + 27))


/*=================================================================*/
/* CDM Info Command Structures */
/*=================================================================*/

typedef struct _wfs_cdm_outpos
{
    WORD             fwPosition;
    WORD             fwShutter;
    WORD             fwOutputPosition;
    WORD             fwTransport;
} WFSCDMOUTPOS, * LPWFSCDMOUTPOS;

typedef struct _wfs_cdm_status
{
    WORD             fwDevice;
    WORD             fwSafeDoor;
    WORD             fwDispenser;
    WORD             fwIntermediateStacker;
    LPWFSCDMOUTPOS * lppOutputPositions;
    LPSTR            lpszExtra;
} WFSCDMSTATUS, * LPWFSCDMSTATUS;

typedef struct _wfs_cdm_caps
{
    WORD            wClass;
    WORD            fwType;
    WORD            wMaxBills;
    WORD            wMaxCoins;
    BOOL            bCompound;
    BOOL            bShutter;
    BOOL            bRetract;
    BOOL            bSafeDoor;
    BOOL            bCoins;
    BOOL            bCylinders;
    BOOL            bCashBox;
    BOOL            bCashIn;
    BOOL            bRefill;
    BOOL            bAutoDeposit;
    BOOL            bVandalCheck;
    BOOL            bIntermediateStacker;
    BOOL            bBillsTakenSensor;
    WORD            fwOutputPositions;
    LPSTR           lpszExtra;
} WFSCDMCAPS, * LPWFSCDMCAPS;

typedef struct _wfs_cdm_physicalcu
{
    LPSTR           lpPhysicalPositionName;
    CHAR            cUnitID[5];
    ULONG           ulCount;
    USHORT          usPStatus;
} WFSCDMPHCU, * LPWFSCDMPHCU;

typedef struct _wfs_cdm_cashunit
{
    USHORT          usNumber;
```

```
    USHORT          usType;
    CHAR            cUnitID[5];
    CHAR            cCurrencyID[3];
    ULONG           ulValues;
    ULONG           ulInitialCount;
    ULONG           ulCount;
    ULONG           ulMinimum;
    ULONG           ulMaximum;
    BOOL            bAppLock;
    BOOL            bDevLock;
    USHORT          usStatus;
    LPSTR           lpPhysicalPositionName;
    USHORT          usNumPhysicalCUs;
    LPWFSCDMPHCU   *lppPhysical;
} WFSCDMCASHUNIT, * LPWFSCDMCASHUNIT;

typedef struct _wfs_cdm_cu_info
{
    USHORT          usTellerID;
    USHORT          usCount;
    LPWFSCDMCASHUNIT *lppList;
} WFSCDMCUINFO, * LPWFSCDMCUINFO;

typedef struct _wfs_cdm_teller_info
{
    USHORT          usTellerID;
    CHAR            cCurrencyID[3];
} WFSCDMTELLERINFO, * LPWFSCDMTELLERINFO;

typedef struct _wfs_cdm_teller_totals
{
    USHORT          usTellerID;
    CHAR            cCurrencyID[3];
    ULONG           ulBills;
    ULONG           ulCoins;
    ULONG           ulCashIn;
    ULONG           ulCashBox;
} WFSCDMTELLERTOTALS, * LPWFSCDMTELLERTOTALS;

typedef struct _wfs_cdm_teller_pos
{
    USHORT          usTellerID;
    USHORT          usPosition;
} WFSCDMTELLERPOS, * LPWFSCDMTELLERPOS;

typedef struct _wfs_cdm_currency_exp
{
    CHAR            cCurrencyID[3];
    SHORT           sExponent;
} WFSCDMCURRENCYEXP, * LPWFSCDMCURRENCYEXP;

typedef struct _wfs_cdm_mix_type
{
    USHORT          usMixNumber;
    USHORT          usMixType;
    USHORT          usSubType;
    LPSTR           lpszName;
} WFSCDMMIXTYPE, * LPWFSCDMMIXTYPE;

typedef struct _wfs_cdm_mix_row
{
    ULONG           ulAmount;
    LPUSHORT        lpusMixture;
} WFSCDMMIXROW, * LPWFSCDMMIXROW;

typedef struct _wfs_cdm_mix_table
{
    USHORT          usMixNumber;
    LPSTR           lpszName;
    USHORT          usRows;
    USHORT          usCols;
    LPULONG         lpulMixHeader;
    LPWFSCDMMIXROW  *lppMixRows;
} WFSCDMMIXTABLE, * LPWFSCDMMIXTABLE;
```

```
typedef struct _wfs_cdm_denomination
{
    CHAR            cCurrencyID[3];
    ULONG           ulAmount;
    USHORT          usCount;
    LPULONG         lpulValues;
    ULONG           ulCashBox;
} WFSCDMDENOMINATION, * LPWFSCDMDENOMINATION;

typedef struct _wfs_cdm_present_status
{
    LPWFSCDMDENOMINATION  lpDenomination;
    WORD                  wPresentState;
    LPSTR                 lpszExtra;
} WFSCDMPRESENTSTATUS, * LPWFSCDMPRESENTSTATUS;


/*=================================================================*/
/* CDM Execute Command Structures */
/*=================================================================*/

typedef struct _wfs_cdm_denominate
{
    USHORT                usTellerID;
    USHORT                usMixNumber;
    LPWFSCDMDENOMINATION  lpDenomination;
} WFSCDMDENOMINATE, * LPWFSCDMDENOMINATE;

typedef struct _wfs_cdm_dispense
{
    USHORT                usTellerID;
    USHORT                usMixNumber;
    USHORT                usPosition;
    BOOL                  bPresent;
    LPWFSCDMDENOMINATION  lpDenomination;
} WFSCDMDISPENSE, * LPWFSCDMDISPENSE;

typedef struct _wfs_cdm_cashin
{
    USHORT                usTellerID;
    LPWFSCDMDENOMINATION  lpDenomination;
} WFSCDMCASHIN, * LPWFSCDMCASHIN;

typedef struct _wfs_cdm_start_ex
{
    USHORT           usTellerID;
    USHORT           usCount;
    LPUSHORT         lpusCUNumList;
} WFSCDMSTARTEX, * LPWFSCDMSTARTEX;

typedef struct _wfs_cdm_calibrate
{
    USHORT        usNumber;
    USHORT        usNumOfBills;
} WFSCDMCALIBRATE, * LPWFSCDMCALIBRATE;

/*=================================================================*/
/* CDM Message Structures */
/*=================================================================*/

typedef struct _wfs_cdm_cu_error
{
    WORD              wFailure;
    LPWFSCDMCASHUNIT  lpCashUnit;
} WFSCDMCUERROR, * LPWFSCDMCUERROR;

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

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

```
#endif  /* __INC_XFSCDM__H */
```