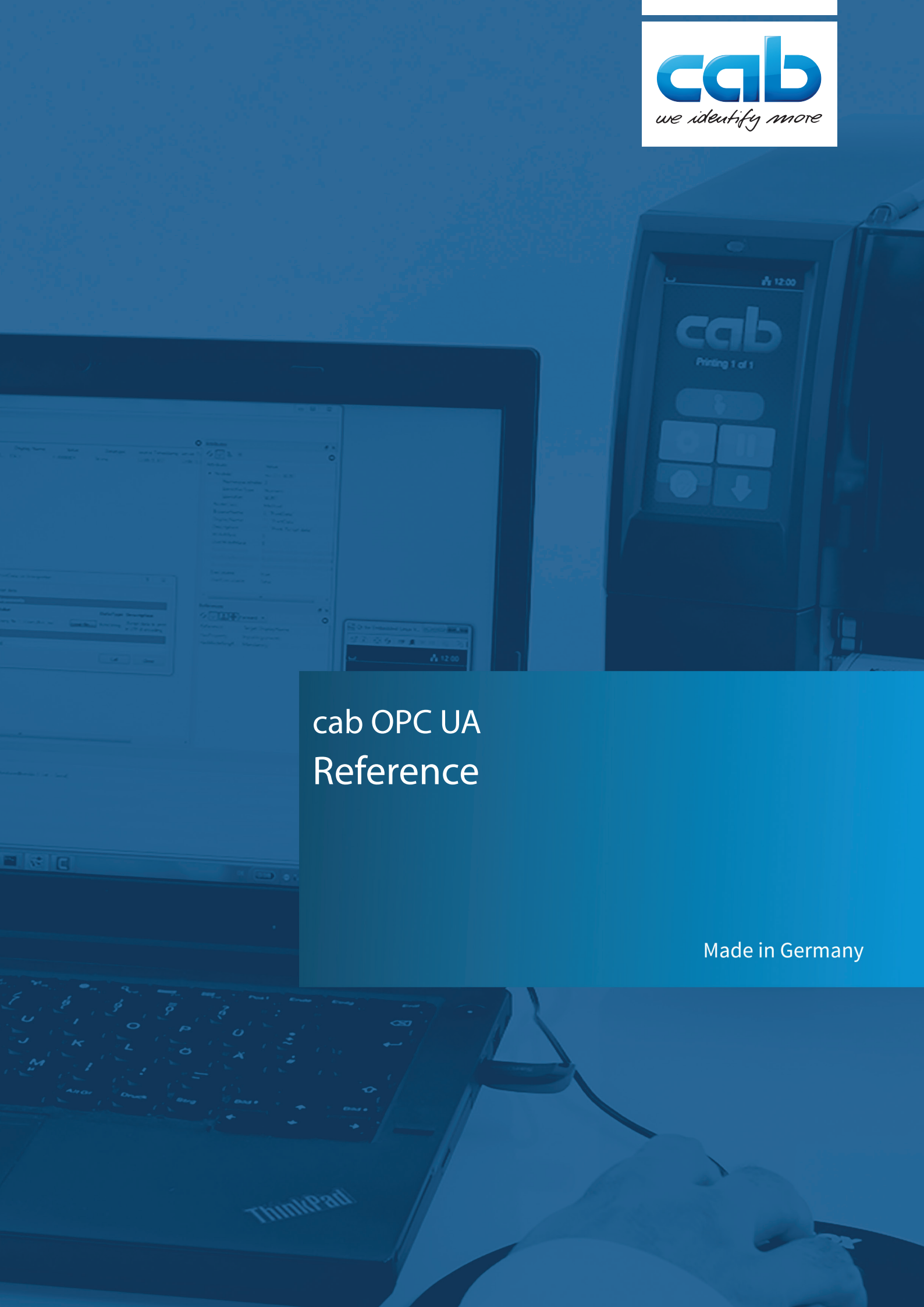

cab OPC UA
Reference

Made in Germany

2 cab OPC UA - Reference

Category	Models
Label printing systems	SQUIX, EOS, MACH 4S, XC Q, XD Q
Print and apply systems	HERMES Q, HERMES QL, AXON 1
Print modules	PX Q

Edition: 07/2026 - **Part No** 9004303

Copyright
This documentation as well as translation hereof are property of cab Produkttechnik GmbH & Co. KG.
The replication, conversion, duplication or divulgement of the whole manual or parts of it for other intentions than its original intended purpose demand the previous written authorization by cab.

Trademark
Microsoft® is a registered trademark of the Microsoft Corporation.
Windows® is a registered trademark of the Microsoft Corporation.
TrueTypeTM is a registered trademark of the Apple Computer, Inc.

Editor
Regarding questions or comments please contact cab Produkttechnik GmbH & Co. KG.

Terms and conditions
Deliveries and performances are effected under the General conditions of sale of cab.

Deutschland
cab Produkttechnik GmbH & Co KG
Karlsruhe
Tel. +49 721 6626 0
www.cab.de

USA
cab Technology, Inc.
Chelmsford, MA
Tel. +1 978 250 8321
www.cab.de/us

Taiwan
cab Technology Co., Ltd.
Taipeh
Tel. +886 (02) 8227 3966
www.cab.de/tw

Singapur
cab Singapore Pte. Ltd.
Singapur
Tel. +65 6701 8691
www.cab.de/en

Frankreich
cab Technologies S.à.r.l.
Niedermodern
Tel. +33 388 722501
www.cab.de/fr

Mexiko
cab Technology, Inc.
Juárez
Tel. +52 656 682 4301
www.cab.de/es

China
cab (Shanghai) Trading Co., Ltd.
Shanghai
Tel. +86 (021) 6236 3161
www.cab.de/cn

Südafrika
cab Technology (Pty) Ltd.
Randburg
Tel. +27 11 886 3580
www.cab.de/za

Table of contents

1 Introduction 4
1.1 Scope 4
1.2 Document structure 4
1.3 Cross references 4

2 Address space map 5
2.1 Namespaces 5
2.2 Object types (overview) 5

3 Object types 6
3.1 PrinterType 6
3.2 PrintHeadType 14
3.3 RibbonSaverType 15
3.4 PeripheralDeviceType 16
3.5 DiagnosticsType 17
3.6 LabelType 19
3.7 ContentFieldsType 19
3.8 SetupValueType 20

4 Object types 21
4.1 Sequence 21
4.2 Methods 22

5 Current label workflow 23
5.1 Workflow 23
5.2 Field names 24

6 Statistics 25
6.1 Where the counters live 25
6.2 Resetting service counters 25

7 Events 26
7.1 Event flow of a print job 26
7.2 ErrorEventType 26
7.3 IOStatusChangeEventType 27
7.4 Event flow of a print job 27
7.5 ErrorEventType 27
7.6 IOStatusChangeEventType 28
7.7 JobCancelEventType 28
7.8 JobFinishEventType 28
7.9 JobInfoEventType 28
7.10 SetupValueChangeEventType 28
7.11 SetupVisibilityChangeEventType 28
7.12 TotalCancelEventType 28

8 Setup parameters 29
8.1 Structure of a setup parameter 29
8.2 Access via NodeId — not via path 29
8.3 Events on changes 30
8.4 Live browsing 30

9 Data types 31
9.1 Enumerations 31
9.2 Structures 33

10 Appendix — NodeId conventions 35
10.1 BrowseName conventions 35

This reference catalogues all object types, methods, events, data types and setup structures of the cab printer OPC UA server.

When you read this document to look something up, the table of contents at the start is your fastest entry point. For the learning path “from commissioning to the first productive print”, see the separate **cab OPC UA — Quick Guide**.

1 Introduction

1.1 Scope

Category	Models
Label printing systems	SQUIX, EOS, MACH 4S, XC Q, XD Q
Print and apply systems	HERMES Q, HERMES QL, AXON 1
Print modules	PX Q

1.2 Document structure

- **Address space map:** namespaces and overview of all object types.
- **Object types:** one section per type — variables, methods, components.
- **Methods:** alphabetical full reference with input and output arguments.
- **Events:** all printer-specific event types with their fields.
- **Setup parameters:** structure and path convention (live-browse the full list).
- **Data types:** printer-specific structures and enumerations.
- **Appendix:** NodeId conventions, ID ranges.

1.3 Cross references

- Companion Quick Guide: `cab_opc_ua_quickguide_en.pdf`
- OPC Foundation Companion Specs: https://reference.opcfoundation.org/
- Address space root: `Objects → DeviceSet → Printer`

2 Address space map

This section gives an overview of the printer OPC UA address space. Detailed descriptions follow in the later chapters.

2.1 Namespaces

The server uses the namespaces listed below. The NS indices may differ slightly depending on the nodeset load order in the server; check the actual order at runtime under `Objects → Server → NamespaceArray`.

NS index	URI	Comment
0	http://opcfoundation.org/UA/	OPC UA Core — standard definitions of the OPC Foundation.
1	http://cab.de/Printer/	cab Printer main namespace — devices, I/O, interpreter, statistics.
2	http://opcfoundation.org/UA/DI/	OPC UA Companion Device Integration — inherited standard properties (Manufacturer, Model, SerialNumber, …).
3	http://cab.de/Printer/Setup/	Setup subtree — device parameters as in the printer menu.
4	http://cab.de/Printer/Content/	Label content — content fields of the currently loaded label.
5	http://cab.de/Printer/Diagnostics/	Diagnostics subtree — logs, loggers and status variables.

2.2 Object types (overview)

Type	Description
`ContentFieldsType`	Represents a content hosting JScript command like T(exts), B(arcodes) or others.
`DiagnosticsType`	Printer diagnostics entry point. Process loggers and log files can be accessed from here.
`LabelType`	Represents a label currently active in the interpreter ready for replace and print commands.
`PeripheralDeviceType`	Base type for all internally or externally attached devices.
`PrintHeadType`	Thermal print head. Depending on printer type, up to two instances are available.
`PrinterType`	cab label printer device type. Base entry point for all printer related information.
`RibbonSaverType`	Internal ribbon saver device. The ribbon saver lifts the print head so, paper can be moved without ribbon.
`SetupValueType`	Base class for all setup configuration types.

Concrete peripheral device types follow in chapter *Object types* under `PeripheralDeviceType`. Event types are listed in their own chapter *Events*.

3 Object types

3.1 PrinterType

cab label printer device type. Base entry point for all printer related information.

Used by instances `Printer`.

3.1.1 Comonents

BrowseName	Type	Description
`ParameterSet`	`BaseObjectType` (OPC UA)	Flat list of Parameters.
`MethodSet`	`BaseObjectType` (OPC UA)	Flat list of Methods.
`I/O`	`FunctionalGroupType` (DI)	Provides software access to the printer's I/O. Generates IOStatusChangeEvent event.
`Statistics`	`FunctionalGroupType` (DI)	Printer usage statistics.
`Ribbon`	`FunctionalGroupType` (DI)	Provides ribbon related information.
`Paper`	`FunctionalGroupType` (DI)	Provides paper related information.
`Interpreter`	`FunctionalGroupType` (DI)	Provides print job status information. Generates ErrorEvent, JobInfoEvent, JobCancelEvent, TotalCancelEvent and JobFinishEvent events.
`USB Key`	`FunctionalGroupType` (DI)	Describes the attached USB service key. Service keys allow to unlock additional features.
`Storage`	`FunctionalGroupType` (DI)	Provides access to the printer's internal and externally attached storage devices.
`Setup`	`FunctionalGroupType` (DI)	Provides read and write access to the printer's configuration.
`SubDevices`	`ConfigurableObjectType` (DI)	Internally or externally attached devices.

3.1.2 Variables

BrowseName	Type	Description
`FPGARevision`	UInt32	FPGA Firmware Revision.

3.1.3 Parameters

Name	Type	Description
`Operating Time`	UInt32	Summarized uptime of printer.
`Labels`	Counter	Number of labels printed.
`Thermal Transfer`	UInt32	Length of printed thermal transfer material.
`Thermal Direct`	UInt32	Length of printed thermal direct material.
`ENDPOS`	Boolean	Applicator is in labeling position. Correlates with the former SUE signal.
`ERROR`	Boolean	The printer is in error state. Correlates with the former SAA signal.
`FEEDON`	Boolean	A feed is currently in progress, i.e. paper is moving. Correlates with the former PTE.
`HOMEPOS`	Boolean	Applicator is in home position. Correlates with the former SOE signal.
`JOBRDY`	Boolean	Print data is available. Correlates with the former EDG signal.
`RIBWARN`	Boolean	The ribbon supply roll diameter has undershot a predefined level. Correlates with the former VWF signal.
`PEELPOS`	Boolean	Behavior without applicator: label is in peel position. With applicators attached, the signal indicates that it's safe now to trigger a START or FSTLBL signal. Correlates with the former ESP signal.
`READY`	Boolean	Printer is ready. Correlates with the former DB signal.
`LBLWARN`	Boolean	Warning end of labels.

RIBERR	Boolean	Printer went out of ribbon.
MEDERR	Boolean	Correlates with the former FWE signal.
SYNC	Boolean	Labels are synchronized.
IO Status	UInt32	Combined I/O output status (bit combination) with the following bits set: RIBWARN=1, ENDPOS=2, FEEDON=4, HOMEPOS=8, PEELPOS=16, JOBRDY=32, ERROR=64, READY=128, LBLWARN=256, RIBERR=512, MEDERR=1024, SYNC=2048. Note that ERROR, RIBWARN, LBLWARN, RIBERR and MEDERR are set active low in this status field. Available bits are hardware dependent.
IO Status IN	Boolean	Combined I/O input status (bit combination) with the following bits set: START=1, REPRINT=2, STOP=4, RSTERR=8, JOBDEL=16, LBLREM=32, PAUSE=64, FSTLBL=128, LBLROT=256, LBLFEED=512, SINGLESTEP=16384.
Active	Boolean	Boolean flag indicating printer is active, i.e. interpreting or printing.
Online	Boolean	Printer is online, i.e. the graphical user interface shows the home screen.
Paused	Boolean	Printer is in PAUSE. Do not mess up with printing on hold as issued by the I/O PAUSE signal.
Printing on hold	Boolean	External PAUSE signal is applied. Printer will not continue printing any further labels until the signal is deasserted.
ESCs	String	String containing the result of JScript's ESCs command.
ESCz	String	String containing the result of JScript's ESCz command.
Last Job	String	Name of last successful job. See the corresponding direct JScript command j.
Labels To Print	UInt32	Labels to be printed within the current job excluding the currently printed label. Note that labels waiting for an external START command are considered to be already in print.
Error	UInt16	Provides detailed error information as in the ESCs command. The numeric value may be interpreted as an ASCII char, e.g. 45 == '-' indicates no error.
abc Status	ABCStatus	Current status of integrated abc basic interpreter, one of Idle, Compiling, Running, RuntimeError or SyntaxError.
Available Space	UInt16	Available queue memory for label processing.
ESCl	String	String containing the result of JScript's ESCl command.
JPhase	JPhase	Current state of JScript interpreter.
Error As String	LocalizedText	Provides translated error information as shown on the printer's display. The individual text entities are separated by RS (ASCII 0x1E).
Ribbon Diameter	UInt32	Detected ribbon diameter.
Ribbon Low	Boolean	Indicates whether the ribbon is considered to be low in reference to the corresponding printer setting.
Paper Diameter	UInt32	Detected paper diameter.
Paper Low	Boolean	Indicates whether the paper is considered to be low in reference to the corresponding printer setting.
Pinch Roller Open	Boolean	Reflects the pinch roller open state (available in selected printer models).
Input Voltage	Double	Printer Input Voltage
Key Serial	String	Serial number of attached cab service key.
Key User Code	String	Admission level of the attached cab service key in string representation, e.g. Service-Key.

IFFS Available Space	UInt64	Available space in kilobytes on internal flash.
IFFS Capacity	UInt64	Capacity in kilobytes of internal flash.
USB Memory Available Space	UInt64	Available space in kilobytes on currently attached USB memory device.
USB Memory Capacity	UInt64	Capacity in kilobytes of currently attached USB memory device.
USB Memory Attached	Boolean	Indicates whether an USB memory device is currently attached to the printer.
SD Card Available Space	UInt64	Available space in kilobytes on currently attached SD Card.
SD Card Capacity	UInt64	Capacity in kilobytes of currently attached SD Card.
SD Card Attached	Boolean	Indicates whether a SD Card is currently attached to the printer.
Temporary Storage Available Space	UInt64	Available space in kilobytes of temporary storage (RAM disk).
Temporary Storage Capacity	UInt64	Capacity in kilobytes of temporary storage (RAM disk).
USER Memory	String	Printer's USER memory contents as accessed by RUSER/WUSER JScript commands. Will be updated only after label was successfully printed.
INF Memory	String	Printer's INF memory contents as accessed by WINF/ESCi JScript commands.
Current Label Definition Completion Sequence	UInt32	A counter incremented for each new job definition completion as provided by JScript A command. May be used to synchronize the jobs fields creation in the interpreter.
Current Label Id	UInt32	A counter incremented for each new job definition received by the interpreter.
Current Label Name	String	Job name of current job of interpreter as provided by JScript 'j' command or auto-generated.
Last Jobs	JobInfo[]	List of finished print jobs, status may be successful or aborted. The array is limited to the 10 newest entries.
Operating Time Total	UInt32	Summarized uptime of printer.
Labels Total	Counter	Number of labels printed.
Thermal Transfer Total	UInt32	Length of printed thermal transfer material.
Thermal Direct Total	UInt32	Length of printed thermal direct material.
Operating Time Service	UInt32	Summarized uptime of printer since last service.
Labels Service	Counter	Number of labels printed since last service.

`Thermal Transfer Service`	UInt32	Length of printed thermal transfer material since last service.
`Thermal Direct Service`	UInt32	Length of printed thermal direct material since last service.

3.1.4 Methods

Alerts

Query for the list of available printer alerts, i.e. error messages.

Output arguments:

Name	Type	Description
`Alert`	Alert[]	Error or informational message.

Bitmap

Returns the last printed label in PNG format as processed by the printengine (i.e. image width is always pritner width).

Output arguments:

Name	Type	Description
`value`	ByteString	PNG print bitmap data.

Possible status codes:

- `BadOutOfMemory`
- `BadResourceUnavailable`
- `BadUnexpectedError`

CancelJob

Cancels the current print job.

ExportSettings

Exports the printer's configuration in XML format. This format can be loaded by the printer via ImportSettings.

Output arguments:

Name	Type	Description
`xml output`	XmlElement	XML configuration data.

FileDelete

Delete given file on a printer's storage device.

Input arguments:

Name	Type	Description
`Storage device`	MemoryCardDevice	Storage device selected.
`Filename`	String	File name without path information.

Possible status codes:

- `BadInternalError`
- `BadInvalidArgument`
- `BadUnexpectedError`

FileDownload

Download e.g. a label format or image file from a printer's storage device in a one-step operation. This is only adequate for small amount of data. For larger files, e.g. fonts, it's recommended to use FileOpen.

Input arguments:

Name	Type	Description
`Storage device`	MemoryCardDevice	Storage device selected.
`Filename`	String	File name without path information.

Output arguments:

Name	Type	Description
`File contents`	ByteString	Contents of requested file.

Possible status codes:

- `BadInternalError`
- `BadInvalidArgument`
- `BadResourceUnavailable`
- `BadUnexpectedError`

FileList

List files on given storage device.

Input arguments:

Name	Type	Description
`Storage device`	MemoryCardDevice	Storage device selected.
`Subfolder`	MemoryCardFolder	One of Fonts, Labels, Images, Miscellaneous, Logs.

Output arguments:

Name	Type	Description
`Filesystem contents`	DirectoryEntry[]	Contents of requested device and directory.

Possible status codes:

- BadInternalError
- BadResourceUnavailable
- BadUnexpectedError

FileOpen

Opens a file on a storage device in the given mode. The transfer itself is done via block Read or Write commands, so this method can be used to transfer files of 'arbitrary' size. Typically, you resolve the Write or Read child from the returned file object node id in order to transfer the data.

Input arguments:

Name	Type	Description
Storage device	MemoryCardDevice	Storage device selected.
Filename	String	File name without path information.
Mode	Byte	Bit 0: open for read, Bit 1: open for write, Bit 2: truncate.

Output arguments:

Name	Type	Description
FileNodeId	NodeId	NodeId allocated for File object.
FileHandle	UInt32	The handle to pass with subsequent Read or Write operations.

Possible status codes:

- BadInternalError
- BadInvalidArgument
- BadNotFound
- BadNotReadable
- BadNotWritable
- BadResourceUnavailable
- BadUnexpectedError

FileUpload

Upload e.g. a label format or image file to a printer's storage device in a one-step operation. This is only adequate for small amount of data. For larger files, e.g. fonts, it's recommended to use FileOpen.

Input arguments:

Name	Type	Description
Storage device	MemoryCardDevice	Storage device selected.
Filename	String	File name without path information.
File contents	ByteString	The file contents.

Possible status codes:

- BadInternalError
- BadInvalidArgument
- BadResourceUnavailable
- BadUnexpectedError

FontList

Query for available fonts.

Output arguments:

Name	Type	Description
Font	Font[]	List of available fonts. Either builtin or loaded via JScript MI FNT; command.

ImportSettings

Imports a XML printer configuration.

Input arguments:

Name	Type	Description
xml input	XmlElement	XML configuration data.

LabelBitmap

Returns the last printed label in PNG format with viewport adopted to job settings.

Output arguments:

Name	Type	Description
value	ByteString	Label data as PNG.

LoadLabel

Loads a label from any attached printer storage device.

Input arguments:

Name	Type	Description
Storage device	MemoryCardDevice	Storage device selected.
Filename	String	Label file name without path information.

Possible status codes:

- BadInternalError
- BadInvalidArgument
- BadResourceUnavailable

PrintCurrentLabel

Make the given number of copies from currently loaded label.

Input arguments:

Name	Type	Description
#labels	UInt32	Number of copies.

PrintData

Prints given JScript or ZPL data directly on the printer. The method expects UTF-8 content encoding.

Input arguments:

Name	Type	Description
`language`	PageDescriptionLanguage	Either JScript or ZPL.
`pagedata`	ByteString	Label data, may contain binary data. It is not recommended to send fonts or large images in the payload of this method. Split into multiple calls or use a file upload instead.

Possible status codes:

- `BadInternalError`

QueryObject

Query for object availability.

Input arguments:

Name	Type	Description
`type`	ObjectType	Object type to query, e.g. Font.
`name`	String	Object name to query.

Output arguments:

Name	Type	Description
`availability`	ObjectType	Availability status for the queried object.

Reboot

Reboots the printer.

RecordToFile

Records I/O signals to any attached printer storage device. Calling with empty file name will stop an active recording.

Input arguments:

Name	Type	Description
`Storage device`	MemoryCardDevice	Storage device selected.
`Filename`	String	File name without path information where the I/O signals are logged to.

ResetPasswords

Resets passwords to factory defaults.

ResetServiceCounter

Reset service counter part of printer statistics. Requires at least Service-Key.

ResetSettings

Resets all settings including passwords, keys and certificates to factory defaults.

Screen

Returns a screenshot of the printer's display in PNG format.

Output arguments:

Name	Type	Description
`value`	ByteString	Current printer display contents as PNG image.

SetInput

Set or unset an I/O input, mostly adequate with PAUSE and LBLROT.

Input arguments:

Name	Type	Description
`signal`	IOSignal	Selected input to toggle.
`enable`	Boolean	Toggle on (true) or off (false).

SetOpcUaClientUrl

Sets an OPC UA servers URL for data retrieval in standalone mode.

Input arguments:

Name	Type	Description
`slot`	UInt32	A value from 0 to 4. Up to five separate slots (i.e. different connections) are configurable.
`url`	String	Complete OPC UA TCP/IP url including user and password.

TotalCancel

Cancels all print jobs scheduled or currently printed by the printer.

TriggerInput

Triggers an I/O input, one of FSTLBL, REPRINT, START, LBLREM, JOBDEL, RSTERR, STOP, LBLFEED or SINGLESTEP. PAUSE and LBLROT is not implemented as trigger, use SetInput instead.

Input arguments:

Name	Type	Description
`signal`	IOSignal	Input event to be triggered.

3.2 PrintHeadType

Thermal print head. Depending on printer type, up to two instances are available.

Used by instances `TPH 1`, `TPH 2`.

3.2.1 Components

BrowseName	Type	Description
`ParameterSet`	`BaseObjectType` (OPC UA)	Flat list of Parameters.
`Statistics`	`FunctionalGroupType` (DI)	Print head usage statistics.

3.2.2 Parameters

Name	Type	Description
Operating Time	UInt32	Summarized uptime of printer.
Labels	Counter	Number of labels printed.
Thermal Transfer	UInt32	Length of printed thermal transfer material.
Thermal Direct	UInt32	Length of printed thermal direct material.
Open	Boolean	Current open state of print head.
Temperature	Double	Current print head temperature

3.3 RibbonSaverType

Internal ribbon saver device. The ribbon saver lifts the print head so, paper can be moved without ribbon.

Used by instances Ribbon Saver, Ribbon Saver 2.

3.3.1 Components

BrowseName	Type	Description
ParameterSet	BaseObjectType (OPC UA)	Flat list of Parameters.
MethodSet	BaseObjectType (OPC UA)	Flat list of Methods.

3.3.2 Parameters

Name	Type	Description
Sync	Boolean	Ribbon Saver is initialized and fully operational.
Head State	RibbonSaverStatus	Current state of the print head, one of Printing or Saving.

3.3.3 Methods

Synchronize

Issues a synchronization command. This is typically done automatically on printer startup. Only relevant for service tasks.

ToggleHead

Toggles the head state up or down. Only relevant for service tasks.

3.4 PeripheralDeviceType

Base type for all internally or externally attached devices.

3.4.1 Components

BrowseName	Type	Description
`Statistics`	`FunctionalGroupType` (DI)	Peripheral device usage statistics.

3.4.2 Concrete peripheral devices

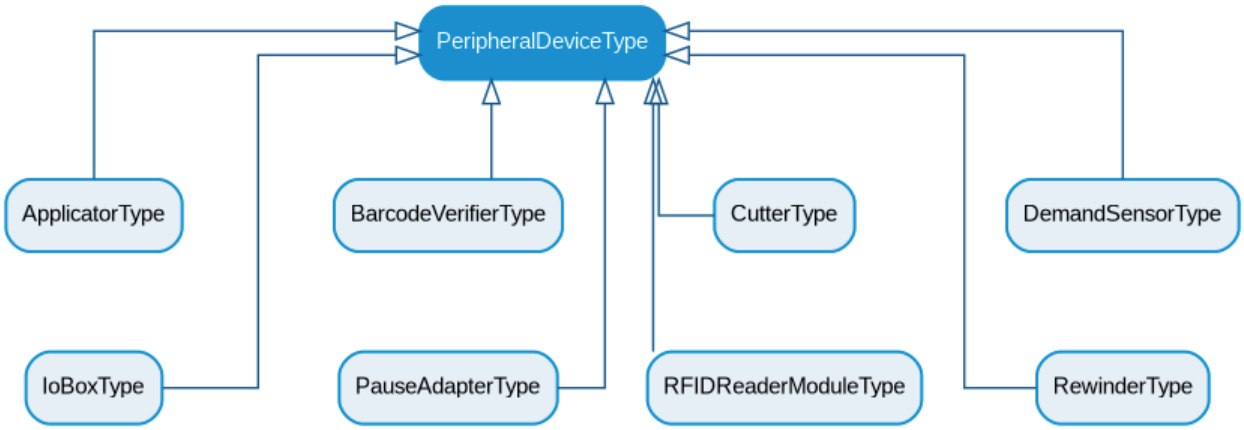

Figure 1 Hierarchy of peripheral device types

The following object types extend `PeripheralDeviceType` with device-specific variables and methods. The structure (Statistics, ResetUserCounters etc.) follows the base type; device-specific details are visible at the instance path in the address space.

Type	Instance in address space	Description
`ApplicatorType`	`Applicator`	Any kind of applicator peripheral device.
`BarcodeVerifierType`	`Barcode Verifier`	Barcode verifier device.
`CutterType`	`Cutter`	Cutter peripheral device.
`DemandSensorType`	`Demand Sensor`	Present sensor peripheral device.
`IoBoxType`	`IoBox`	Input/Output box.
`PauseAdapterType`	`Pause Adapter`	Pause adapter peripheral device.
`RFIDReaderModuleType`	`RFIDReaderModule`	Internal RFID reader module.
`RewinderType`	`Rewinder`	Rewinder peripheral device.

3.4.3 Methods of `RFIDReaderModuleType`

ResetUserCounters

Zero's RFID statistics user counters.

3.5 DiagnosticsType

Printer diagnostics entry point. Process loggers and log files can be accessed from here.

Used by instances `Diagnostics`.

3.5.1 Components

BrowseName	Type	Description
`Logs`	`FolderType` (OPC UA)	Provides a collection of essential log files.
`Loggers`	`FolderType` (OPC UA)	Allows configuration of logger objects for all printer tasks.

3.5.2 Variables

BrowseName	Type	Description
`USB`	String	Status of attached USB devices.
`Network`	String	Status of all network devices and routing tables.
`StatusVariables`	String	Internal printer status. Only available with attached USB key of appropriate level.
`Open Files`	String	List of open files.
`Processes`	String	List of active processes.

3.5.3 Methods

Close

Available on `SystemLog`, `KernelLog`, `WPALog`, `DataStream`.

Input arguments:

Name	Type	Description
`FileHandle`	UInt32	

Possible status codes:

- `BadInternalError`

GetPosition

Available on `SystemLog`, `KernelLog`, `WPALog`, `DataStream`.

Input arguments:

Name	Type	Description
`FileHandle`	UInt32	

Output arguments:

Name	Type	Description
`Position`	UInt64	

Possible status codes:

- `BadInternalError`
- `BadInvalidArgument`

Open

Available on SystemLog, KernelLog, WPALog, DataStream.

Input arguments:

Name	Type	Description
Mode	Byte	

Output arguments:

Name	Type	Description
FileHandle	UInt32	

Possible status codes:

- BadInternalError
- BadInvalidArgument
- BadNotFound
- BadNotReadable
- BadNotWritable
- BadResourceUnavailable
- BadUnexpectedError

Read

Available on SystemLog, KernelLog, WPALog, DataStream.

Input arguments:

Name	Type	Description
FileHandle	UInt32	
Length	Int32	

Output arguments:

Name	Type	Description
Data	ByteString	

Possible status codes:

- BadInternalError
- BadInvalidArgument

SetPosition

Available on SystemLog, KernelLog, WPALog, DataStream.

Input arguments:

Name	Type	Description
FileHandle	UInt32	
Position	UInt64	

Possible status codes:

- `BadInternalError`
- `BadInvalidArgument`

Sync

Sync log levels to be available after printer reboot.

Available on `Drivers`.

3.6 LabelType

Represents a label currently active in the interpreter ready for replace and print commands.

3.6.1 Components

BrowseName	Type	Description
`Content Fields`	`ContentFieldsType`	List of available content fields in current label.

3.6.2 Variables

BrowseName	Type	Description
`Id`	UInt32	A counter incremented for each new job definition received by the interpreter.
`Name`	String	Job name of current job of interpreter as provided by JScript 'j' command or auto-generated.

3.7 ContentFieldsType

Represents a content hosting JScript command like T(exts), B(arcodes) or others.

Used by instances `Content Fields`.

3.7.1 Possible content types

A `ContentFieldsType` container holds variables whose concrete type is one of the subtypes of `ContentFieldType`:

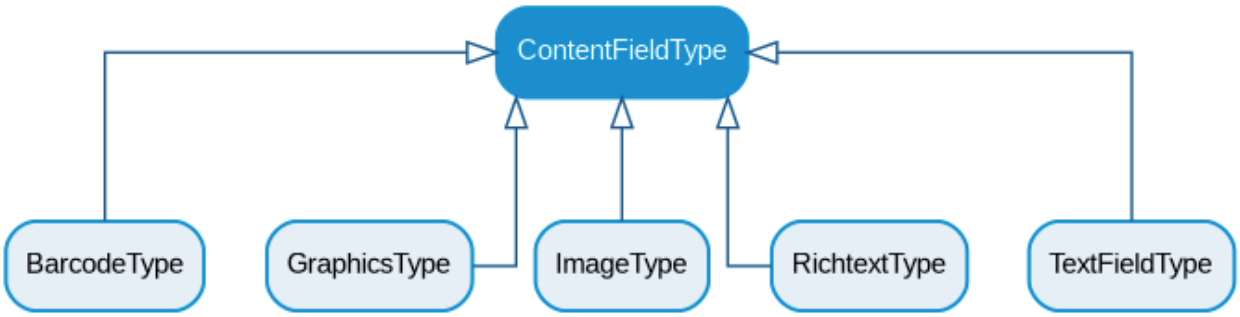

Figure 2 Hierarchy ContentFieldType

Type	Description
`BarcodeType`	Barcode field as described by printer language.
`GraphicsType`	Graphics field as described by printer language.
`ImageType`	Image as described by printer language.
`RichtextType`	Richtext field as described by printer language.
`TextFieldType`	Text field as described by printer language.

`ContentFieldType` is abstract: Represents a content hosting JScript command like T(exts) or B(arcodes).

3.8 SetupValueType

`SetupValueType` is the abstract base type for all setup parameters. Every concrete parameter in the address space is an instance of one of the subtypes shown below. Which additional sub-components a parameter carries is documented in chapter *Setup parameters*.

3.8.1 Subtypes

Base class for all setup configuration types.

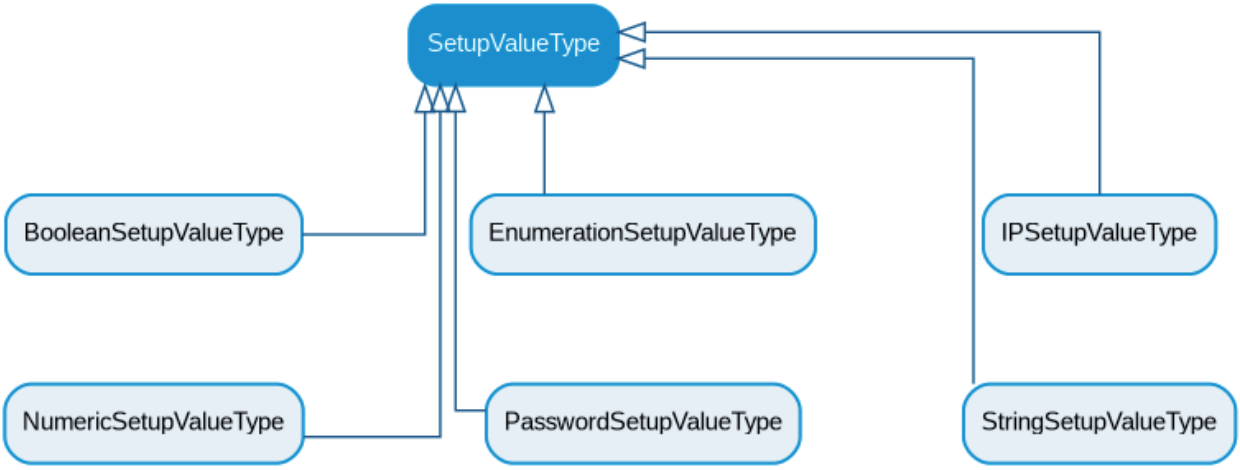

Figure 3 Hierarchy SetupValueType

Type	Description
`BooleanSetupValueType`	A boolean setup value. Stores a simple boolean value, i.e. true or false.
`EnumerationSetupValueType`	A list setup value. Each list entry is a name/value pair.
`IPSetupValueType`	IPv4 setup value.
`NumericSetupValueType`	A numerical setup value. Stores a numerical value including range and stepping information.
`PasswordSetupValueType`	A password setup value. Stores a password string and input check parameters. Returned always as masked characters.
`StringSetupValueType`	A string setup value. Stores a string and input check parameters.

4 Firmware update

The printer supports remote firmware updates over OPC UA via the *Temporary File Transfer* pattern. A client uploads the firmware image chunk-wise into a server-allocated temporary file and triggers the update by calling `CloseAndCommit`. The printer validates the image, switches to the new firmware and reboots automatically.

4.1 Sequence

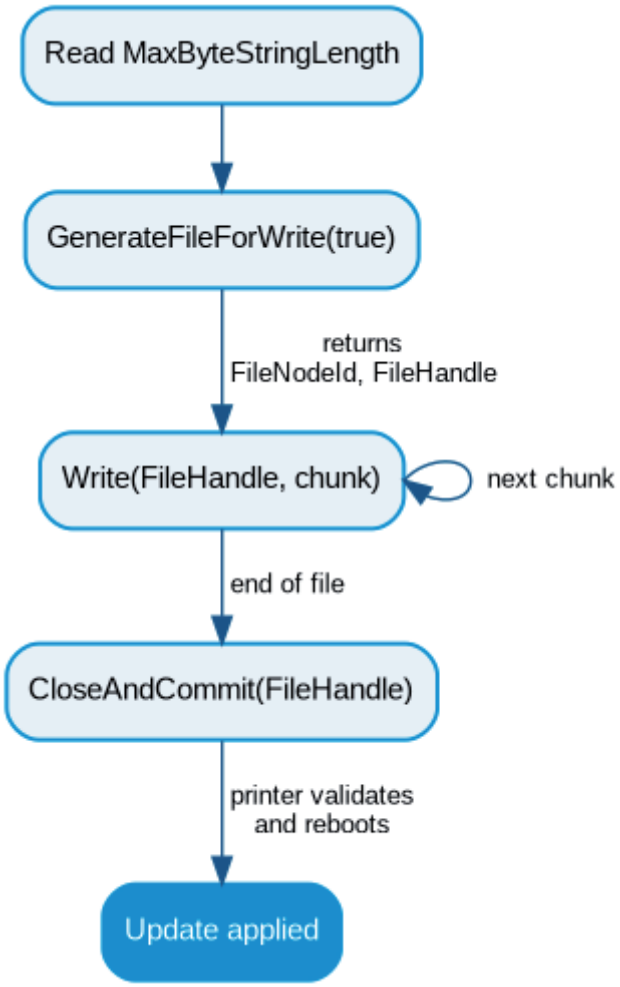

Figure 4 Firmware update sequence

1. **Read the maximum chunk size** from the `Server → ServerCapabilities → MaxByteStringLength` variable. Each subsequent `Write` call must not exceed this size.
2. **Open a temporary file for writing** by calling `GenerateFileForWrite` on the printer's `TemporaryFileTransferType` object with `GenerateOptions = true`. The method returns a `FileNodeId` (a temporary `FileType` instance on the server) and a `FileHandle` (UInt32) for use in the upload loop.
3. **Upload the firmware image** by repeatedly calling the standard OPC UA `Write` method on the returned file node, passing the handle and the next chunk of file bytes. Continue until the entire image has been transferred.
4. **Finalize the upload and start the update** by calling `CloseAndCommit` on the `TemporaryFileTransferType` object with the `FileHandle`. The printer validates the image, switches to the new firmware and reboots. `CloseAndCommit` returns the `NodeId` of a `CompletionStateMachine` that the client can subscribe to for fine-grained progress, or the client may simply wait for the printer to reconnect.

4.2 Methods

4.2.1 GenerateFileForWrite

Main entry point for initiating a firmware upload.

Input arguments:

Name	Type	Description
GenerateOptions	i=24	Undocumented. Just set to ‘true’.

Output arguments:

Name	Type	Description
FileNodeId	NodeId	NodeId allocated for File object.
FileHandle	UInt32	The handle to pass with subsequent Write operations.

4.2.2 CloseAndCommit

Starts firmware update on printer.

Input arguments:

Name	Type	Description
FileHandle	UInt32	The handle as retrieved from GenerateFileForWrite.

Output arguments:

Name	Type	Description
CompletionStateMachine	NodeId	Unused.

5 Current label workflow

A label sent to the printer is treated as a **template**: the interpreter parses the layout but does not print yet. The OPC UA server exposes the prepared template as the **Current Label** object so a client can read the content-field tree that the interpreter has built, substitute field values and issue the print explicitly. Two methods bring a template into this state:

- `LoadLabel(storage, filename)` — loads a template file from the printer's storage. Files can be placed there by direct upload (USB stick, SD card) or via the printer's built-in WebDAV virtual device.
- `PrintData(language, pagedata)` — sends the template as JScript or ZPL data directly with the call.

By convention the template ends with the JScript `A[NO]` command so the printer only **prepares** the label and does not start printing automatically once the template has been parsed. The client decides when to print by issuing a separate `PrintCurrentLabel` call after the field values have been substituted.

5.1 Workflow

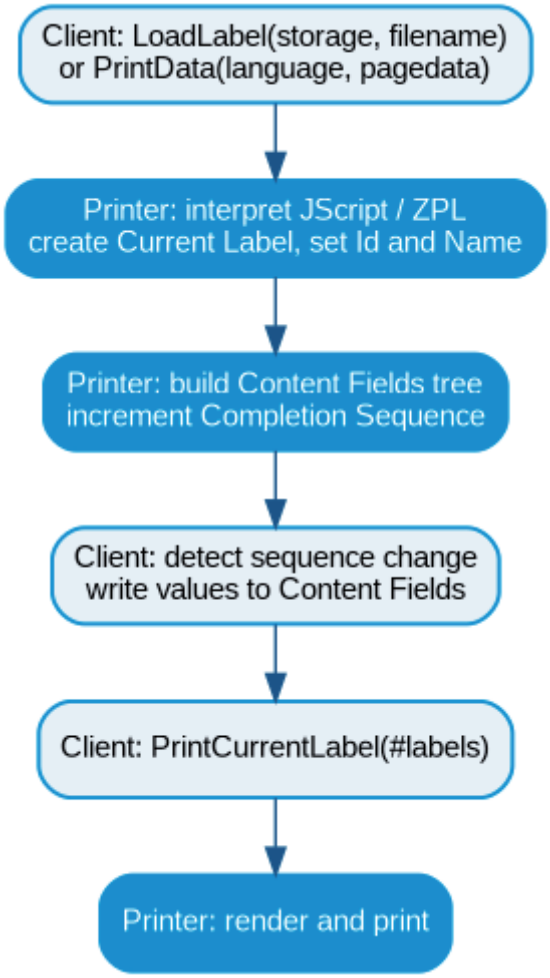

Figure 5 Current label workflow

1. **Send the template** via `LoadLabel` or `PrintData`. The interpreter parses the JScript/ZPL content, creates the *Current Label* object and updates the PrinterType variables `Current Label Id` (an incrementing counter) and `Current Label Name` (job name from the JScript `j` command or auto-generated).
2. **Wait for the content fields to be available.** When the interpreter has finished building the field tree of the label (triggered by the JScript `A` command), it increments `Current Label Definition Completion Sequence`. The client monitors this counter to know that all `Content Fields` sub-nodes now exist and are ready to be written.

3. **Set field contents** by writing the desired value to each field under `Current Label → Content Fields`. The concrete field types (Text, Barcode, …) are listed under `ContentFieldType` in the *Object types* chapter; each subtype exposes its own value variable. Content-field variables are **write-only**: a client can substitute values but cannot read back the current contents.
4. **Trigger printing** with `PrintCurrentLabel(#labels)`. The printer renders and prints the prepared label the requested number of times.

Without the `A[NO]` convention at the end of the label definition the printer would start printing immediately after step 1 with the default content of each field — usually not the desired behaviour when the client wants to substitute values.

Method details for `LoadLabel`, `PrintData` and `PrintCurrentLabel` are documented in the *Object types* chapter under `PrinterType`.

5.2 Field names

The BrowseName of a sub-node under `Content Fields` is the JScript field name assigned in the template. In a JScript command the name is the token between the command letter and the first parameter list, separated by a colon — for example `T:HEADLINE;10,10,0,3,5;Initial text` defines a text field with the name `HEADLINE`, which appears in the address space as `Current Label → Content Fields → HEADLINE`. The same convention applies to barcodes (`B:CODE;…`) and other label-format commands. The example below shows a minimal template with one text and one barcode field and how the named fields appear in the address space:

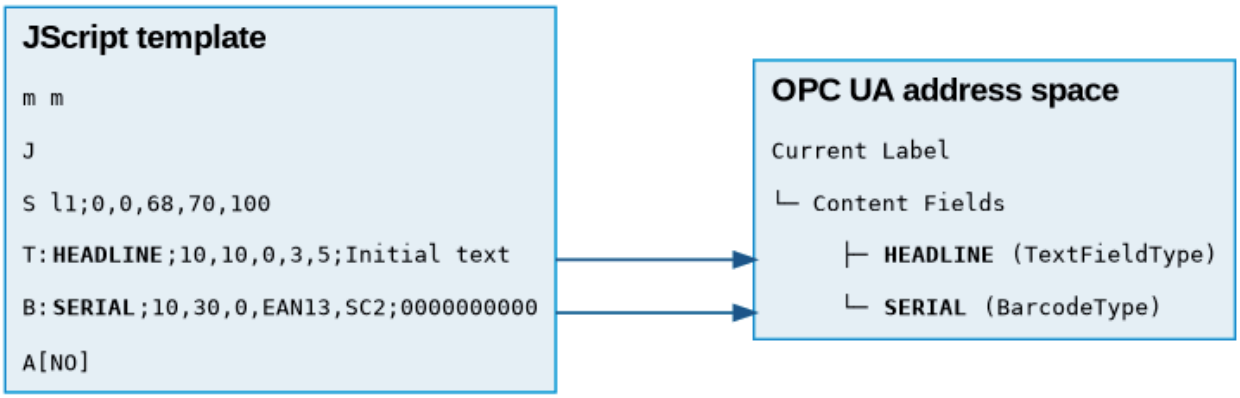

Figure 6 JScript field mapping

Field names must follow JScript rules:

- alphanumeric only (no special characters),
- must start with a letter,
- case-sensitive,
- unique within the template — duplicate names are rejected.

Only fields with an explicit JScript name are reachable from OPC UA. Anonymous fields exist on the printout but cannot be addressed by a client.

6 Statistics

The printer exposes operational counters under `Statistics` folders attached to several object types. Each counter is kept in two flavours:

- **Total** — cumulative since manufacture; cannot be reset.
- **Service** — cumulative since the last service reset; can be cleared with the `ResetServiceCounter` method.

The two flavours sit in dedicated sub-folders named `Total` and `Service` underneath each `Statistics` node.

6.1 Where the counters live

Statistics folders exist at three levels of the address space:

Location	Counter list documented
`Printer → Statistics`	Yes — see chapter Object types, `PrinterType`
`Printer → PrintHead → Statistics`	Yes — see chapter Object types, `PrintHeadType`
`Printer → SubDevices → <Device> → Statistics`	No — counters are reported dynamically by each peripheral at runtime

Peripheral devices that publish statistics describe their counters themselves; the printer aggregates whatever the device reports. The set of counters and their meaning therefore depends on the connected peripheral — to discover what a specific device exposes, browse its `Statistics` folder on a running printer.

6.2 Resetting service counters

Only the `Service` part of a Statistics tree can be reset. The Total counters are immutable.

- `ResetServiceCounter` on the printer's `MethodSet` clears the Service counters under `Printer → Statistics → Service`. Requires at least Service-Key.
- Peripheral devices that support service counters expose their own `ResetServiceCounter` method on the device's `MethodSet`. The call clears the Service counters of that device only.

`ResetUserCounters` on `RFIDReaderModuleType` is a separate method — it clears user-controlled counters that are not part of the Service/Total split.

7 Events

This section lists all printer-specific event types that the OPC UA server sends to subscribed clients. Each type is a subclass of `BaseEventType`; the inherited mandatory fields (`EventId`, `EventType`, `SourceNode`, `Time`, `Message`, `Severity`) are not listed.

7.1 Event flow of a print job

When the print engine processes a print job it emits one `JobInfoEvent` per label and a closing `JobFinishEvent` once the engine has returned to idle. If a fault occurs (paper jam, cutter blocked, …) an `ErrorEvent` interrupts the regular flow.

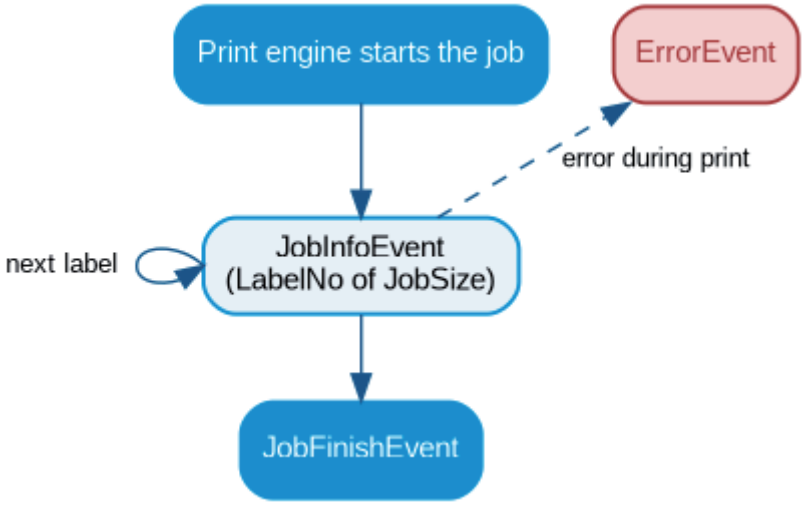

Figure 7 Print-job event sequence

- `JobInfoEvent` — fired for each label about to be printed. Carries `LabelNo` and `JobSize` so a client can track progress through a multi-label job (e.g. `#labels > 1` in `PrintCurrentLabel` or multiple `A`-commands in JScript).
- `JobFinishEvent` — fired once the engine has returned to idle. From a client perspective the job is complete.
- `ErrorEvent` — emitted on any fault during processing. See `ErrorEventType` below for the printer-specific semantics of `Severity` and `Message`.

I/O status transitions (`JOBREADY`, `FEEDON`, `PEELPOS`, …) are also reported as `IOStatusChangeEvent`s but are not synchronised with the `JobInfoEvent`/`JobFinishEvent` boundaries — they reflect the printer's electrical I/O state, which a client interested in job progress does not need to interpret.

7.2 ErrorEventType

Error message from printer.

Fields:

Name	Type	Description
`Detail1`	LocalizedText	Detailed error information part 1.
`Detail2`	LocalizedText	Detailed error information part 2.
`ErrorType`	SByte	Type of error as provided by JScript command ESCs.

Inherited fields with printer-specific semantics:

- `Message` (LocalizedText) — the main error message text, assembled from the printer's error code and details. Localised according to the printer's UI language.
- `Severity` (UInt16) — `200` if the event is an informational message (`msg->code == ,-'`), not an actual error; `900` for system errors (`msg->code == ,s'`); `800` for any other error, e.g. cutter blocked.
- `SourceName` (String) — describes the emitter of the error message.

7.3 IOStatusChangeEventType

This section lists all printer-specific event types that the OPC UA server sends to subscribed clients. Each type is a subclass of `BaseEventType`; the inherited mandatory fields (`EventId`, `EventType`, `SourceNode`, `Time`, `Message`, `Severity`) are not listed.

7.4 Event flow of a print job

When the print engine processes a print job it emits one `JobInfoEvent` per label and a closing `JobFinishEvent` once the engine has returned to idle. If a fault occurs (paper jam, cutter blocked, …) an `ErrorEvent` interrupts the regular flow.

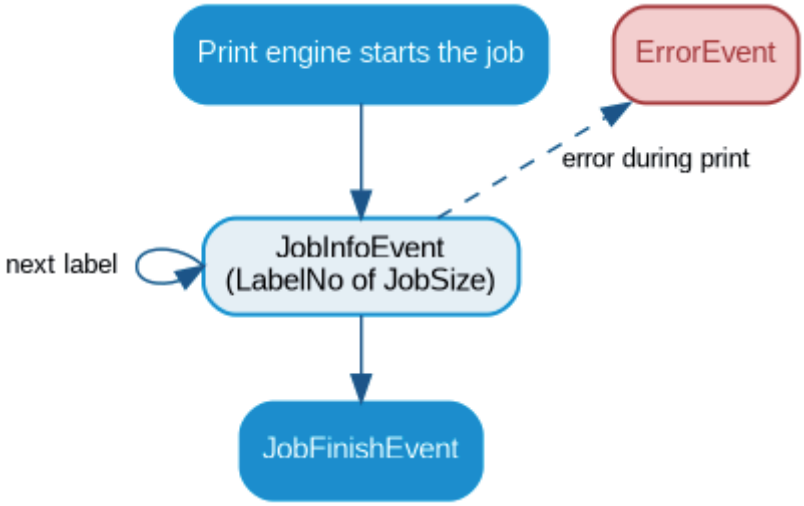

Figure 8 Print-job event sequence

- `JobInfoEvent` — fired for each label about to be printed. Carries `LabelNo` and `JobSize` so a client can track progress through a multi-label job (e.g. `#labels > 1` in `PrintCurrentLabel` or multiple `A`-commands in JScript).
- `JobFinishEvent` — fired once the engine has returned to idle. From a client perspective the job is complete.
- `ErrorEvent` — emitted on any fault during processing. See `ErrorEventType` below for the printer-specific semantics of `Severity` and `Message`.

I/O status transitions (`JOBREADY`, `FEEDON`, `PEELPOS`, …) are also reported as `IOStatusChangeEvent`s but are not synchronised with the `JobInfoEvent`/`JobFinishEvent` boundaries — they reflect the printer's electrical I/O state, which a client interested in job progress does not need to interpret.

7.5 ErrorEventType

Error message from printer.

Fields:

Name	Type	Description
`Detail1`	LocalizedText	Detailed error information part 1.
`Detail2`	LocalizedText	Detailed error information part 2.
`ErrorType`	SByte	Type of error as provided by JScript command ESCs.

Inherited fields with printer-specific semantics:

- `Message` (LocalizedText) — the main error message text, assembled from the printer's error code and details. Localised according to the printer's UI language.
- `Severity` (UInt16) — `200` if the event is an informational message (`msg->code == ,-'`), not an actual error; `900` for system errors (`msg->code == ,s'`); `800` for any other error, e.g. cutter blocked.
- `SourceName` (String) — describes the emitter of the error message.

7.6 IOStatusChangeEventType

I/O input or output status changed.

Fields:

Name	Type	Description
Status	UInt32	Combined I/O output status (bit combination). See parameter I/O Status.
Status IN	UInt32	Combined I/O input status (bit combination). See parameter I/O Status IN. The new input status will be evaluated by the printer firmware. For tracking purpose only.

7.7 JobCancelEventType

A (sub-)job as defined by JScript command j was cancelled on the printer.

Fields:

Name	Type	Description
JobId	String	JobId as defined by JScript command j or an empty string in case of a TotalCancel.

7.8 JobFinishEventType

Finish message from printer. Printer returned to idle state.

7.9 JobInfoEventType

Job information message from printer. Will be emitted whenever a label is going to be printed.

Fields:

Name	Type	Description
IsTestPrint	Boolean	Indicates whether currently printed label is a test print.
JobSize	UInt32	Number of labels in current print job.
LabelNo	UInt32	Number of label under print in current print job.
JobId	String	JobId as defined by JScript command j.
IsPseudoPrint	Boolean	TRUE for previews and non-printing pages.

7.10 SetupValueChangeEventType

A distinct setup value has changed.

7.11 SetupVisibilityChangeEventType

A distinct setup value has changed its visibility flag due to a configuration change. Example: warn level ribbon is hidden as soon as transfer print is selected. The OPC UA setup tree though does not remove the tree entry itself but sets the IsActive flag appropriately.

7.12 TotalCancelEventType

A total cancel was carried out on the printer. Can be used for synchronization purpose. Once this event is received, it’s safe to send new print data.

8 Setup parameters

The setup parameters live in the address space below …/`Printer/Setup` and mirror the layout of the printer touchscreen menu.

8.1 Structure of a setup parameter

Each setup parameter is a variable whose concrete type is one of the sub-types of `SetupValueType` (see chapter *Setup value types*). Which sub-components a parameter carries depends on this type. The parameter value itself is always at the parameter node and is read or written via `Read`/`Write`.

All setup parameters share the property

Sub-component	Type	Meaning
`IsActive`	Boolean	currently visible / available (depends e.g. on attached peripherals or service level)

Additional sub-components per type:

8.1.1 NumericSetupValueType

Sub-component	Type	Meaning
`Range`	Range	minimum and maximum of the allowed value
`Step`	depends on data type	step size
`Unit`	LocalizedText	engineering unit (mm, dB, °C, …)
`ZeroValue`	depends on data type	internal representation of “zero/off”

8.1.2 StringSetupValueType

Sub-component	Type	Meaning
`MinLen`	UInt32	minimum length
`MaxLen`	UInt32	maximum length
`Regex`	String	regular expression for input validation

8.1.3 PasswordSetupValueType

Sub-component	Type	Meaning
`MinLen`	UInt32	minimum length
`MaxLen`	UInt32	maximum length

The value itself is returned masked; writes are only possible on encrypted endpoints.

8.1.4 BooleanSetupValueType, EnumerationSetupValueType, IPSetupValueType

No further sub-components — the value itself carries all the information. For `EnumerationSetupValueType` the value is an integer selection out of an externally defined list.

8.2 Access via NodeId — not via path

The hierarchical path of a parameter below …/`Printer/Setup` follows the structure of the printer menu. These browse names **may change** between firmware revisions, printer models or as a result of internal reorganisation.

Therefore access setup parameters **always via the unique NodeId**, never via the browse name path. NodeIds remain stable and are the only reliable address. Determine them once by browsing in UaExpert or programmatically via the `TranslateBrowsePathsToNodeIds` service, then cache them in your client and use the NodeId directly for all subsequent reads and writes.

8.3 Events on changes

- `SetupValueChangeEvent` — the value of a setup parameter changed.
- `SetupVisibilityChangeEvent` — the visibility (`IsActive`) of a setup parameter changed (e.g. because a peripheral was attached or configured).

Both event types are described in detail in chapter *Events*.

8.4 Live browsing

The concrete parameter list is not printed statically in this reference — it changes with printer model, firmware version and attached peripherals and would go stale quickly in a static document.

Recommendation: browse in UaExpert starting from `Objects → DeviceSet → Printer → Setup`. For a versioned snapshot of the current state, traverse the Setup subtree from your own OPC UA client and write the result into an inventory format of your own.

9 Data types

This chapter lists the printer-specific structures and enumerations used in method arguments and variables of the address space.

9.1 Enumerations

9.1.1 ABCStatus

Value	Name	Meaning
67	Compiling	abc program is being compiled.
69	RuntimeError	An exception has occurred during the execution of an abc program.
73	Idle	No abc program currently executed.
82	Running	An abc program is currently executed.
83	SyntaxError	The transferred abc program data isn't correct.

9.1.2 FontType

Value	Name	Meaning
1	Bitmap	Pixel-based bitmap font.
4	TrueType	Scalable TrueType outline font.

9.1.3 IOSignal

Value	Name	Meaning
0	FSTLBL	Print first label. Only for Cycle sequence = Apply-Print.
1	REPRINT	The last printed label will be repeated.
2	START	Print start signal. Only for Print on demand = On.
3	LBLREM	Confirmation of the superior control that the label has been taken from the peel-off position.
4	JOBDEL	The current print job is canceled and deleted from the print buffer.
5	RSTERR	Error state of the printer will be reset.
6	STOP	Stop signal to interrupt the operation.
7	LBLFEED	Send feed operation. Availability is hardware dependent.
8	PAUSE	Set labeling on hold. PAUSE is not a trigger signal, use this with ToogleInput method.
9	LBLROT	Turn label 90° for applicators with selection of label orientation. LBLROT is not a trigger signal, use this with ToogleInput method.
10	SINGLESTEP	Applicator required. Applicator steps once in its internal state machine.

9.1.4 JPhase

Execution state of builtin JScript interpreter.

Value	Name	Meaning
0	WaitingForLabelDefinition	Interpreter in idle state.
1	InProcessOfLabelDefinition	Interpreter has received a job start command but job definition is not complete.
2	Printing	Deprecated.
3	CompleteLabelDefinitionAvailable	Job definition is complete. Interpreter is ready to receive new job data. Label fields may be accessed under the interpreter's subnode 'Current Label'.

9.1.5 JobStatus

Value	Name	Meaning
0	FinishedSuccess	The print job completed successfully.
1	Aborted	The print job was aborted before completion.

9.1.6 LogLevel

Value	Name	Meaning
-1	off	No log message generated.
0	emergency	Printer cannot continue operation. Reboot required.
1	alert	A severe error occurred. Immediate service intervention required.
2	critical	A critical error occurred. User or service action required.
3	error	An error occurred. User action required.
4	warning	An exceptional event occurred, but printer operation can continue.
5	notice	A less common informational message occurred.
6	info	Informational message. This is the default level.
7	debug	A debugging message. Enabling debugging messages could have a negative impact on printer performance.

9.1.7 MemoryCardDevice

Value	Name	Meaning
0	Default Storage	The storage marked as default, either of IFFS, USB Memory, SD Card or WEBDAV Drive.
1	IFFS	Internal flash file system.
2	USB Memory	USB memory device.
3	SD Card	Secure Digital Memory Card.
4	WebDAV Drive	WebDAV Drive.
5	Temporary Storage	Temporary Storage (RAM Disk).

9.1.8 MemoryCardFolder

Value	Name	Meaning
0	Fonts	Folder for any kind of font.
1	Images	Folder for any kind of image.
2	Labels	Folder for any kind of label (JScript, ZPL), as .prn, .lbl or .zpl.
3	Miscellaneous	Folder for database, USER or TMP files.
4	Logs	Folder for any kind of log file generated on user request, e.g. save system log.

9.1.9 ObjectAvailability

Value	Name	Meaning
0	Available	Requested object is available in system memory.
1	NotAvailable	Requested object not available, neither in system memory, nor on external storage device.
2	OnMemoryCardDevice	Requested object is available on external storage device.

9.1.10 ObjectType

Value	Name	Meaning
0	BitmapFont	Object is a Bitmap font.
1	ScalableFont	Object is a TrueType font.
2	DataBase	Object is a database file, i.e. SQLITE3 or dBase.
3	MediaFile	Object is a media (FMT) file.
4	Image	Object is an image file.
5	Label	Object is a label file.

9.1.11 PageDescriptionLanguage

Value	Name	Meaning
0	JScript	cab printer language.
1	ZPL	Zebra Programming Language.

9.1.12 RibbonSaverStatus

Value	Name	Meaning
0	Printing	Print head is lowered position, ribbon is moving.
1	Saving	Print head in raised position, ribbon is not moving.

9.2 Structures

9.2.1 Alert

Field	Type	Description
Message	String	Contents of alert message.
Timestamp	DateTime	Time of occurrence.

9.2.2 DirectoryEntry

Field	Type	Description
Name	String	Name of directory entry.
Size	UInt32	Size of directory entry in bytes.
Modified	DateTime	Last modification date of directory entry.

9.2.3 ExtendedJobInfo

Field	Type	Description
Id	UInt32	A counter incremented for each new job definition received by the interpreter.
Name	String	Job name of current job of interpreter as provided by JScript ‘j’ command or auto-generated.
Status	JobStatus	
Timestamp	DateTime	
HeatLevel	Int32	Effective heatlevel as calculated from GUI settings and label configuration.
PrintSpeed	Int32	Effective printspeed during print in [mm/s].
OffsetX	Double	Effective X offset during print in [mm].
OffsetY	Double	Effective Y offset during print in [mm].

9.2.4 Font

Field	Type	Description
`Id`	Int32	JScript identifier for the font.
`Type`	FontType	Either a Bitmap or TrueType Font.
`Name`	String	The font's name.
`Description`	String	The font's description.
`Builtin`	Boolean	Indicates whether the font is provided by the firmware or loaded from an external source.

9.2.5 JobInfo

Field	Type	Description
`Id`	UInt32	A counter incremented for each new job definition received by the interpreter.
`Name`	String	Job name of current job of interpreter as provided by JScript 'j' command or auto-generated.
`Status`	JobStatus	Job completion status.
`Timestamp`	DateTime	Job completion timestamp.

10 Appendix — NodeId conventions

The namespaces are documented in the chapter *Address space map* as a table.

10.1 BrowseName conventions

- Object types carry the suffix `Type` (`PrinterType`, `ApplicatorType`).
- Event types carry the suffix `EventType` (`JobFinishEventType`).
- Methods carry a verb-style name (`LoadLabel`, `FileList`, `TriggerInput`).
- Variables with read/write semantics have descriptive noun names (`Paper Diameter`, `Ribbon Low`, `JPhase`).