EPSON®
EXCEED YOUR VISION

ePOS Fiscal Print Solution

Development Guide

Notes

Trademarks

Epson Italia S.p.A.
Via Margherita Viganò De Vizzi, n. 93/95
20092 Cinisello Balsamo (MI)
Italy

REVISION HISTORY

Rev	Sheet	Comment	Date	Author
		Previous revisions available on request		
P	Various	queryContentByDate and queryContentByNumbers line limits updated. setDate year updated. Can be either 2 or 4 digits long. query string timeout field updated. directIO timeout attribute added. printerCommands file type added. EFT POS ERROR message added to error table. authorizeSales response added. Automatic Programming chapter added. Added FP NO ANSWER NETWORK error.	20th Dec 2016	P. Barnett
Q	Various	Added RT status request (new attribute). Added commercial refund document and commercial void document. Added RT examples. Document updated throughout.	13th Mar 2018	P. Barnett
R	Various	Updated commercial refund and void documents. Updated queryPrinterStatus. Updated and expanded examples.	29TH Mar 2019	P. Barnett
S	Various	Corrected printerCommands encapsulation in 3.6 Commands Structure chapter. Updated examples.	23rd Sep 2020	P. Barnett
T	All	Fiscal receipt references renamed to commercial document (old scontrino fiscale term) Administrative document and non-fiscal receipt references renamed to Management document (old scontrino non-fiscale term). Removed free invoice as no longer supported. Added Native Command Cross Reference chapter. Added new examples including login. Added the following commands: • EFTPOSDailyClosure • EFTPOSGetCurrentTotal • printRecLotteryID • printRecRefundAdjustment • rebootWebServer Added modifiers (deposit, free of charge and single-use voucher). Document updated throughout.	5th Jul 2021	P. Barnett

Index

1 Overview

1.1 Description

The aim of this manual is to provide information to software designers who want to interface Epson fiscal and non-fiscal TM printers (latterly via a fiscal printer) using the fiscal ePOS-Print solution.

1.2 List of Related Documents

- Epson FP Manuale Operatore
- Active X / OCX user manual
- Communication Protocol
- Protocollo di Comunicazione NDA Biglietteria
- Intelligent Printer Printing System ePOS-Print Technical Data
- Gestione Fatture
- FP Gestione Logo
- Fiscal Printer Intelligent Features Guide

1.3 Abbreviations

- DGFE: Old Italian acronym for EJ

- EJ: Electronic Journal
  In Italy, was known as the Dispositivo Giornale di Fondo Elettronico (DGFE) but now is known as the Memoria Permanente di Dettaglio (MPD)

- FP: Fiscal Printer
  In Italy, known as a Registratore Telematico (RT)

- MF: Misuratore Fiscale (obsolete term) or Memoria Fiscale (Fiscal Memory)

- MPD: New Italian acronym for EJ

- MPR: New Italian acronym for Memoria Fiscale (Fiscal Memory)

- RT: Registratore Telematico (Telematic fiscal printer)

1.4 Structure

The Fiscal ePOS-Print solution consists of three components:

- FpMate CGI service
- Fiscal-ePOS-Print XML
- Fiscal ePOS-Print API

1.4.1 FpMate CGI Service

The service is essentially an XML parser which resides inside fiscal printers that reads and interprets fiscal ePOS-Print XML for printing, reporting, configuration and reading.

1.4.2 Fiscal ePOS-Print XML

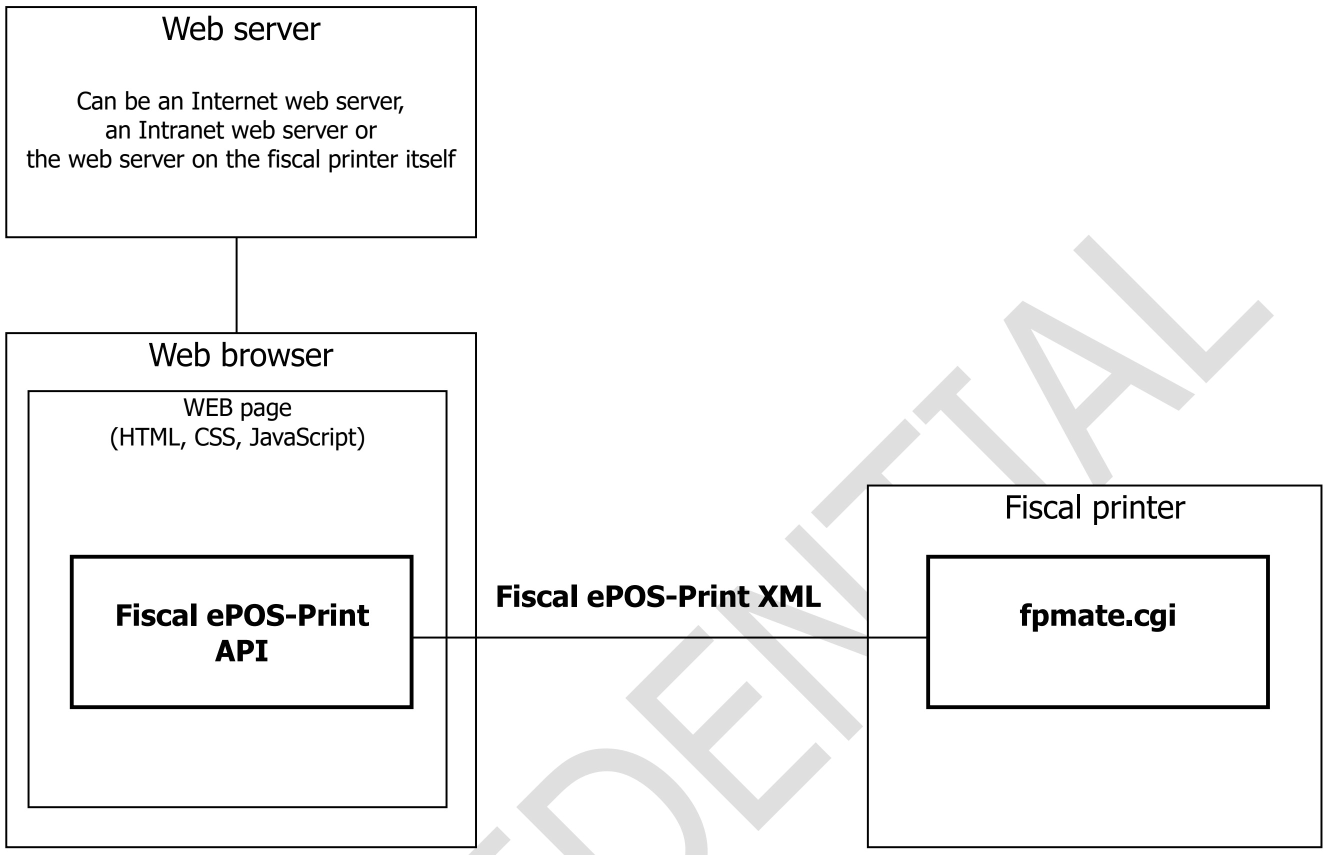
This encompasses the XML command system that defines the main fiscal printer functions in XML. All XML elements and attributes are described in this document.

1.4.3 Fiscal ePOS-Print API

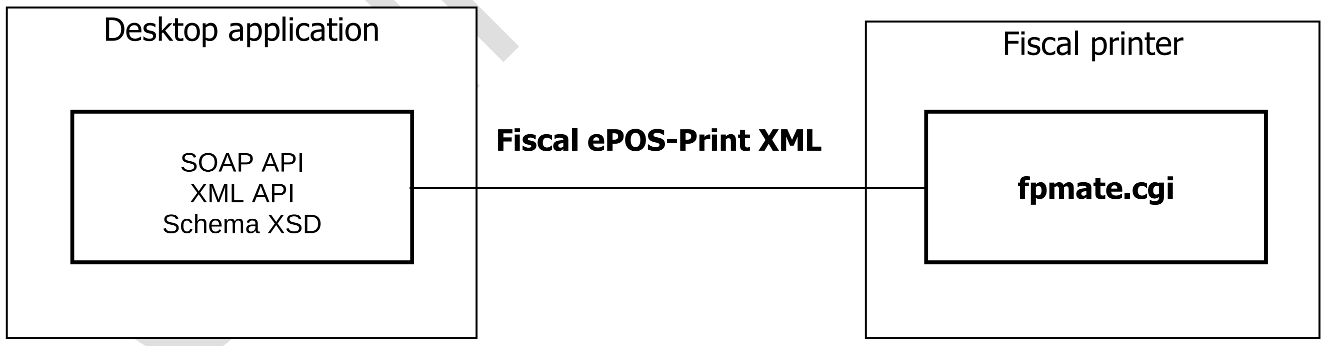
The API is a basic application programming interface that forwards requests to the fiscal ePOS-Print service by means of client-side JavaScript code.

1.5 Typical Topologies

1.5.1 Web Application


Web server
Can be an Internet web server, an Intranet web server or the web server on the fiscal printer itself
Web browser
WEB page (HTML, CSS, JavaScript)
Fiscal ePOS-Print API
Fiscal ePOS-Print XML
Fiscal printer
fpmate.cgi

1.5.2 Desktop Application



2 FpMate CGI Service

2.1 Features

The FpMate CGI service is an XML web service incorporated into the fiscal printer that uses fiscal ePOS-Print XML for printing, reporting, configuration and reading. When a document request is sent from a host to the FpMate CGI service via SOAP/HTTP, it processes the document and afterwards sends back a response document. Printing is possible from various devices and operating systems that support communication via SOAP/HTTP. HTTPS is also supported. The key features and benefits are:

- Installation of drivers and plug-ins is not required
- No PCs or servers are required for printing
- Adds printing functionality to public and private clouds
- Uses TM printer built-in fonts
- Manages fiscal printers
- Manages TM printers via fiscal printers
- Supports Server Direct Print and Remote Request Job functions

2.2 Specification

2.2.1 The Basic End Point Address

The URL format is as follows:

http://[host]/cgi-bin/fpmate.cgi

A timeout value can be added as a query string entry as follows:

http://[host]/cgi-bin/fpmate.cgi?timeout=[timeout]

When the fiscal printer forwards and manages the request destined for another fiscal printer or a non-fiscal TM printer, a query string is appended as follows:

http://[host]/cgi-bin/fpmate.cgi?devid=[device ID]

The two can be combined in the following manner:

http://[host]/cgi-bin/fpmate.cgi?devid=[device ID]&timeout=[timeout]

[host]
Specifies the IP address or domain name of the fiscal printer where the FpMate CGI service resides.

[device ID]
Destination printer selection by name. The device ID of the fiscal or non-fiscal TM printer is specified here. The specific printer that the XML is ultimately destined for is denoted by its device ID which is passed in the query string that forms part of the URL. A null or "local_printer" device ID indicates the use of the same printer as where the web server resides, i.e. the same as in the host portion of the URL. ePOS-print requests destined for non-fiscal TM printers must contain a device ID. Please see the FpMate CGI Configuration section regarding the configuration and programming of device ID entries.

[timeout]
The optional timeout field can be used with:

- authorizeSales EFT-POS payment command. Default 30 seconds
- directIO calls. Default six seconds
- printRecMessage. Default eight seconds
- Secondary printer management
- To indicate the timeout to use when the FpMate CGI service begins communication with the fiscal firmware. Default 10 seconds

The value is expressed in milliseconds. The alternative timeout attribute if present in the XML overrides the query string setting in some cases. When the specified timeout period elapses, the operation is cancelled. However, XML lines beforehand may have been processed normally.

Examples

```
http://192.168.192.168/cgi-bin/fpmate.cgi?devid=local_printer&timeout=12000
```

An example URL for sending a request for onward printing is as follows:

```
http//192.168.1.1/cgi-bin/fpmate.cgi?devid=cucina
```

If the device does not exist an error is returned. Bad field names are ignored (e.g. DEVID/devID/DEV££ID/TIMEOUT). Timeout values with non-numeric characters are also ignored.

2.2.2 Request Message Format

A SOAP message forms part of the HTTP POST request method. A single document request is specified in the SOAP body.

[Example]

```
POST /cgi-bin/fpmate.cgi?devid=local_printer&timeout=10000 HTTP/1.1
Content-Type: text/xml; charset=utf-8
Content-Length: xxxx
If-Modified-Since: Thu, 01 Jan 1970 00:00:00 GMT

<?xml version="1.0" encoding="utf-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
    <s:Body>
            <printerCommand>
                    <queryPrinterStatus  operator="" statusType="" />
            </printerCommand>
    </s:Body>
</s:Envelope>
```

2.2.3 Response Message Format

A SOAP message is returned as the response to the HTTP POST method. The SOAP body contains the response document. To permit cross-domain communication, the response contains the HTTP "Access-Control-Allow-Origin: *" header. See *Fiscal Printer Intelligent Features Guide* for browser cross-domain information.

[Example]

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: text/xml; charset=utf-8
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
    <s:Body>
            <response success="true" code="" status="xxxxx" />
    </s:Body>
</s:Envelope>
```

3 Fiscal ePOS-Print XML

The fiscal ePOS-Print XML is a printer command system that defines the main fiscal functions in XML.

This XML is intended to facilitate the control of printers, independent of hardware or operating systems, from personal computers and POS terminals, as well as from network devices such as web application servers, smartphones and tablet computers.

The XML structure must contain one of the following root elements as it identifies the type of request. Root elements (also known as root nodes) have empty tags with no attributes or data. Currently none of the XML in the entire library contains any data except for addInfo (additional information in response messages), printGraphicCoupon and setLogo (where base64 binary data is inserted); everything else is passed as tags and attributes.

The last line must contain the root element closure statement. For example:

```
</printerFiscalReceipt>
```

Root Element Tag Name	**Description**
printerFiscalReceipt	Commercial document
printerFiscalReport	Daily closure (X and Z reports)
printerNonFiscal	Management document
printerFiscalDocument	Invoice
printerTicket	Box office ticket (known as a titolo d'accesso)
printerCommand	Command (e.g. printer status)
printerCommands	Multiple requests and programming
epos-print	Print to non-fiscal TM printer (see related documentation)
fpMateConfiguration	Configure or read secondary devices table
eposDevice	For future use

3.1 XML Hierarchy

There are two types of fiscal ePOS-Print XML documents:

- Request documents
- Response documents

The documents are subdivided into two main categories:

- Multi-command: Commercial document, Management document, Invoice, "Printer Commands" and ePOS-Print to non-fiscal TM printer
- Single command: Report, "Printer Command" and FpMate configuration

A printer receives a message from host that contains a document and after having processed the document, sends back a message (to the same host) that contains a response document.

The tables below illustrate the main fiscal root elements and their associated sub-elements. Some of the sub-elements can be used with more than one of these root elements such as directIO and openDrawer. Most of the commercial document sub-elements can also be used with direct invoices that will be explained later.

Commercial Document	**Management Document**	**Report**
printerFiscalReceipt	**printerNonFiscal**	**printerFiscalReport**
beginFiscalReceipt	beginNonFiscal	printXReport
printRecItem	printBarCode	printZReport
printRecItemVoid	printGraphicCoupon	printXZReport
printRecItemAdjustment	printNormal	
printRecItemAdjustmentVoid	printRecMessage	
printRecMessage	setLogo	
printRecRefund *	endNonFiscal	
printRecRefundVoid		
printRecRefundAdjustment		
printRecLotteryID		
printRecSubtotal		
printRecSubtotalAdjustment		
printRecSubtotalAdjustVoid		
printRecTotal		
printGraphicCoupon		
printBarCode		
endFiscalReceipt		

* Refunds are converted to corrections (storni) on RT commercial sale documents.

Command	**Commands**	**Fiscal Document**	**Box Office Tickets**
printerCommand	**printerCommands**	**printerFiscalDocument**	**printerTicket**
authorizeSales	beginTraining	beginFiscalDocument	enterTicket
beginTraining	endTraining	All the printerFiscalReceipt commands apart from printRecLotteryID and printGraphicCoupon	beginTicket
endTraining	printContentByDate	endFiscalReceipt / endFiscalDocument	printTicket
EFTPOSDailyClosure	printContentByNumbers		printTicketVoid
EFTPOSGetCurrentTotal	printDuplicateReceipt		printBarCode
getDate	printRecCash		endTicket
printContentByDate	printRecVoid		exitTicket
printContentByNumbers	resetPrinter		
printDuplicateReceipt	setDate		
printRecCash	setLogo		
printRecVoid			
queryContentByDate			
queryContentByNumbers			
queryPrinterStatus			
rebootWebServer			
resetPrinter			
setDate			
setLogo			

The table below illustrates the common commands that can be used in multi-command documents or as the single command in a printerCommand document:

openDrawer
directIO
clearText
displayText

The table below illustrates the configuration, ePOS-print and Response root elements with their associated sub-elements.

Configuration	**ePOS Print (TM printer)**	**Response**
fpMateConfiguration	**epos-print**	**response**
addDevice	barcode	addInfo
readDevices	cut	
readVersion	feed	
	text	
	Refer to ePOS document for complete command set	

3.2 Commercial Document Structure

Commercial documents must contain the following structure:

```
<printerFiscalReceipt>
        <beginFiscalReceipt  operator="" />
                Sale(s)
                Payment(s)
        <endFiscalReceipt  operator="" />
</printerFiscalReceipt>
```

Responses are sent after "endFiscalReceipt" has been processed. Specific commercial document responses are described in the chapter entitled 5.4.1 Commercial Document Response File.

3.3 Management Document Structure

Management documents must contain the following structure:

```
<printerNonFiscal>
        <beginNonFiscal  operator="" />
                Commands such as for printing text, barcodes, logos or coupons
        <endNonFiscal  operator="" />
</printerNonFiscal>
```

Responses are sent after "endNonFiscal" has been processed. Specific management document responses are described in the chapter entitled 5.7.1 Management Document Response File.

3.4 Report Structure

This structure is used to print out reports (Z report, X report etc.). Only one of the sub-elements should be present here since the FpMate CGI service manages just one report command at a time.

```
<printerFiscalReport>
        <printZReport  operator="" timeout="" />
</printerFiscalReport>
```

Responses arrive after the reports have been printed and if relevant whenever the RT data transmission has been completed. Specific report responses are described in the chapter entitled 5.8.1 Report Response File.

3.5 Command Structure

The key difference here is that the structure must only contain one command. For example:

```
<printerCommand>
        <getDate />
</printerCommand>
```

Responses arrive immediately or after printing. Additional information is not always present since it depends on the command. Please see the specific command.

3.6 Commands Structure

This structure is used if you wish to send multiple command type elements, For example, multiple Direct IO requests:

```
<printerCommands>
        <directIO  command="4015" data="09001" />
        <directIO  command="4015" data="10002" />
</printerCommands>
```

Responses are sent after the last command has been processed and indicates the printer status.

3.7 Fiscal Document Structure

The structure depends on the type of document. Please see the document types table (chapter 5.11.2 Document Types).

The following is required:

```
<printerFiscalDocument>
        <beginFiscalDocument  operator="" documentType="" … />
        -----
        -----
Either  <endFiscalReceipt  operator="" />or
        <endFiscalDocument  operator="" /> or
        Nothing in the case of invoices based on the last commercial document
</printerFiscalDocument>
```

endFiscalDocument or endFiscalReceipt is used with direct invoices. The operationType attribute is no longer valid with invoices.

Responses are sent after either "endFiscalDocument" or "endFiscalReceipt" has been processed or immediately in the case of invoices based on the last commercial document. Specific fiscal document responses are described in chapter 5.11.1 Fiscal Document Response File.

3.8 Box Office Ticket Structure

This structure is used to print out "Titoli d'Accesso" (box office tickets) in Italy. The structure must be as follows:

```
<printerTicket>
        <enterTicket  randomNum="" controlCode="" />
                <beginTicket  randomNum="" controlCode="" />
                <printTicket  randomNum="" controlCode="" textType="R" font="1" text="TEST" />
                <printTicket  randomNum="" controlCode="" textType="R" font="2" text="TEST" />
                -----
                -----
                <printTicket  randomNum="" controlCode="" textType="S" font="4" text="SEAL" />
                <printBarCode  operator="1" position="0" width="2" height="120" hRIPosition="1"
                hRIFont="A" codeType="CODE128" code="{BAFB7F9A37AD2CD64" />
                <endTicket  randomNum="" controlCode="" />
        <exitTicket  randomNum="" controlCode="" />
</printerTicket>
```

Tickets can also be cancelled:

```
<printerTicket>
        <printTicketVoid  randomNum="" controlCode="" />
</printerTicket>
```

Responses are sent after the last command has been processed. Specific box office ticket responses are described in the chapter entitled 5.12.1 Ticket Response File.

3.9 Configuration Structure

This structure is used to configure the FpMate CGI service so that it can manage secondary devices (the so-called Fiscal Printer Hub Function). There is no limit to the number of devices that can be added. It is also used to determine the FpMate CGI service version.

```
<fpMateConfiguration>
    <addDevice  deviceId="Cucina" deviceIp="192.168.1.44" deviceType="non-fiscal"
    deviceModel="TM-T70" deviceRetry="10000" />
</fpMateConfiguration>

<fpMateConfiguration>
    <readVersion />
</fpMateConfiguration>
```

3.10 ePOS-Print Structure (for TM printer only)

The structure follows the same standard as for non-fiscal printers. Additional information is not presented (see the ePOS-Print_eng.pdf document). This function is seldom used because most TM printers now accept SOAP requests directly.

3.11 Response Structure

Responses always contain the following default structure:

```
<response success="true" code="" status="2">
        <addInfo>
                <elementList>lastCommand</elementList>
                <lastCommand>YY</lastCommand>
        </addInfo>
</response>
```

Further information is reflected in additional elements whose names are also appended to the comma delimited elementList data. This varies from command to command.

lastCommand indicates the last executed native fiscal command. Expressed as a number – See chapter 10 Native Command Cross Reference and Communication Protocol document for reference. Either one to four digits long with no leading zeroes. For example:

- 74 = 1-074 printer status
- 80 = 1-080 printRecItem
- 88 = 1-088 printer reset
- 138 = 1-138 printer RT status
- 278 = 1-278 EFT-POS lines
- 6543 = Invalid command

lastCommand can also indicate the last internally generated command.

3.12 Malformed Structures

Failure to adhere to the structure can have unforeseen results. Common errors are:

- <endFiscalReceipt /> missing – The commercial document remains open and an "INCOMPLETE FILE" response will be returned to the browser

- Insufficient or no payment in commercial document – endFiscalReceipt will return an error

- </printerFiscalReceipt> missing – A parser error is immediately returned

- Bad attribute value – The printer processes and prints up until the sub-element with the erroneous line after which a printer error is returned

3.13 Printer Anomalies

To counter any printer anomalies, the FpMate CGI service automatically performs a printer reset every time a fiscal/non-fiscal/report/document request arrives. This means for example that if a previous commercial document was open at the time the request arrives it will be cancelled so that a new one can be printed.

4 Fiscal ePOS-Print XML Details (Configuration)

4.1 FpMateConfiguration

This element is used to determine the FpMate CGI service embedded version or to configure and read secondary devices. The latter is only necessary when the "Hub" function is employed which allows the "master" printer to control secondary fiscal/non-fiscal TM printers.

By virtue of networking, the FpMate CGI service can manage as many printers as you like both fiscal and non-fiscal. Non-fiscal TM printers are not sent requests directly – the fiscal printer acts as a proxy. Whenever this functionality is used, the following must be configured for each secondary device prior to use:

- Device ID
- IP Address (the TCP port is fixed at 9100)
- Printer type (fiscal or non-fiscal)
- Printer model
- Retry

The information is saved internally in the fiscal printer. This is not required if the fiscal printer operates "stand-alone". Non-fiscal TM printer requests destined for a fiscal device and vice-versa are refused and an appropriate error is returned. To setup the fiscal printer to manage secondary printers, an appropriate one-time configuration write request needs to be sent. This overwrites any configuration previously performed so if the previous details need to be maintained, a configuration read request must be performed beforehand. The configuration is not retained in the case of a printer restart.

4.1.1 Root Elements

The root elements for configuration XML file types are:

- fpMateConfiguration
- Response

4.1.2 Sub-elements

4.1.2.1 addDevice

When writing the configuration file, first the addDevice sub-element is used for each distinct device and it can be repeated as many times as you like. It has the following attributes:

N	Attribute	Description
1	deviceId	Device identifier – Can be anything you like (but avoid operating system environment variable names such as HOME or PATH)
2	deviceIp	IP address of the device in numerical not DNS format
3	deviceType	TmPrinter or FpPrinter for example
4	deviceModel	FP-90 or TM-T70 for example
5	deviceRetry	Specifies the retry interval towards the secondary printer, in milliseconds

Responses contain the following structure with no additional information:

```
<response success="true" code="" status="xxxxxx">
</response>
```

4.1.2.2 readDevices

Read requests always have the following structure with no attributes:

```
<fpMateConfiguration>
        <readDevices />
</fpMateConfiguration>
```

Responses are presented as additional information in the following format:

```
    <elementList>deviceCount</elementList>
    <deviceCount>xx
        <deviceNumber1  deviceId="Cucina" deviceIp="192.168.1.44" deviceType="TmPrinter"
        deviceModel="TM-T70" deviceRetry="10000" />
        <deviceNumber2  deviceId="Servizio" deviceIp="192.168.1.45" deviceType="FpPrinter"
        deviceModel="FP-90" deviceRetry="10000" />
        And so on
     </deviceCount>
```

It is also possible to view the stored file called fpMateConfiguration.xml directly from your browser:

- http://xxx.xxx.xxx.xxx/cgi-bin/fpMateConfiguration.xml

4.1.2.3 readVersion

This sub-element is used to determine the FpMate CGI service embedded version. It has the following structure with no attributes:

```
<fpMateConfiguration>
        <readVersion />
</fpMateConfiguration>
```

The response is presented as additional information in the following format:

```
        <elementList>fpmateVersion</elementList>
                <fpmateVersion></fpmateVersion>
```

4.1.3 Adding Devices Example

The following example sets up two secondary printers:

```
<fpMateConfiguration>
    <addDevice  deviceId="Cucina" deviceIp="192.168.1.44" deviceType="TmPrinter"
    deviceModel="TM-T70" deviceRetry="10000" />
    <addDevice  deviceId="Servizio" deviceIp="192.168.1.45" deviceType="FpPrinter"
    deviceModel="FP-90" deviceRetry="10000" />
</fpMateConfiguration>
```

5 Fiscal ePOS-Print XML Details (Printer Documents and Commands)

5.1 Description

In this section, root elements are described. The root element identifies the type of file. This allows for the control of any kind of print out and the use of various commands.

First, the common attributes and sub-elements are explained. Then every root element, its sub-elements and attributes will be described in detail.

5.2 Common Attributes

Many of the attributes passed to fiscal printers are common to many sub-elements. The following table illustrates them:

N	Attribute	Description
1	operator	Operator ID number (range 1 to 12). The operator attribute can be omitted. In this case, Operator 01 is assumed. A +50 offset can used with printRecItem, printRecRefund and printRecItemVoid (regarding printRecItemVoid, the offset cannot be used if voidLastItem="1"). Normally if the quantity = 1, the quantity line with "1x" is not printed. Adding the offset to the operator attribute instructs the fiscal printer to print the line when the quantity is exactly 1. The offset has no effect with direct invoices
2	description	String whose maximum length varies according to the print-out. Currently the maximum is: • Commercial documents – 38 characters • Direct invoices – 37 characters When printing invoices based on the last commercial document, any 38-character descriptions are truncated to 37 characters. Descriptions that exceed limits are truncated. Note that other attribute names with a similar meaning can have higher maximum lengths such as the "data" attribute used with the "printNormal" sub-element
3	department	Department ID number (range 1 to 99)
4	quantity	Epson fiscal printers can compute quantities from 0.001 up to 9999.999. The FpMate CGI service automatically rounds down quantities with more than three decimal places. If the quantity exceeds 9999.999, an error is returned. Either a comma or a full stop (period) can represent the decimal point. Thousand separators should not be used. The value must not be zero
5	unitPrice amount payment	Epson fiscal printers can accept prices from 0.00 up to 9999999.99. The FpMate CGI service automatically rounds down amounts with more than two decimal places. If it exceeds 9999999.99, an error is returned. Either a comma or a full stop (period) can represent the decimal point. Thousand separators should not be used. The unitPrice and payment attributes can be zero, but the amount attribute cannot be zero
6	justification	Set the customer display description justification. The maximum number of characters that can appear on the display on one line is 20, but descriptions can be longer. This parameter tells the fiscal printer whether to take the first 20 characters or the last 20 from the relative description text to use on the display. Set the attribute as follows: 1 = First 20 characters 2 = Last 20 characters

N	Attribute	Description
7	font	Fiscal printers support four different font types. Set the attribute as follows: 1 = Normal 2 = Bold 3 = Double height 4 = Bold and double height The font attribute used with barcodes has a different meaning. Please see the ePOS-Print_eng document for an explanation of non-fiscal TM printer fonts
8	timeout	Expressed in milliseconds with the following elements: • directIO • printZReport • printXZReport • printRecMessage
9	comment	Ignored by the printer. Useful especially for directIO requests

5.3 Common Sub-elements

The following table illustrates the common fiscal printer sub-elements that can be used irrespective of the fiscal printer root element (request type). However, they cannot be used with ePOS-print non-fiscal TM printer requests.

N	Sub-element	Description
1	openDrawer	Open the cash drawer
2	directIO	Send PDU strings (allows full use of all the fiscal printer functionality)
3	clearText	Blank the display
4	displayText	Send messages to the display

5.3.1 openDrawer

Opens the cash drawer. This sub-element has only one attribute to indicate the operator.

```
<openDrawer  operator="" />
```

5.3.2 directIO

Sends PDU native communication protocol strings to the fiscal printer. directIO gives the added benefit of being able to send any one of the supported command strings to the printer so you can make full use of all the fiscal printer functionality. Two attributes are passed – command and data. A "timeout" attribute can also be optionally passed. The default timeout is six seconds. This overrides an eventual query string timeout. Please refer to the "Communication Protocol" document for the command reference. The command attribute is a concatenation of H1 and H2 and is always four bytes long. The data attribute contains the remaining fields and its length varies depending on the command. The timeout is expressed in milliseconds. Care must be taken that the protocol is followed to the letter. None of the corrective mechanisms (such as rounding and truncating) operate here. The following example opens the cash drawer, defines operator 01 and sets an eight second timeout.

```
<directIO  command="1050" data="01" timeout="8000" comment="Open cash drawer, operator 1 and five second timeout" />
```

DirectIO responses contain the following additional information:

- lastCommand – See 3.11 Response Structure
- printerStatus – Five-byte status code. See Responses section for full explanation
- responseCommand – Direct IO command. Should be four bytes
- responseData – Contains the fiscal protocol PDU string returned by the fiscal printer (from the point after the four-byte command up to but not including the checksum)

An example response message:

```
<response success="true" code="" status="x">
  <addInfo>
    <elementList>lastCommand,printerStatus,responseCommand,responseData</elementList>
    <lastCommand>74</lastCommand>
    <printerStatus>20010</printerStatus>
    <responseCommand>1047</responseCommand>
    <responseData>0123071218130001</responseData>
  </addInfo>
</response>
```

5.3.3 clearText

Blanks the display. This sub-element has only one attribute to indicate the operator.

```
<clearText  operator="" />
```

5.3.4 displayText

Sends text messages to the customer display. You cannot insert carriage returns or line feeds so use spaces to pad out line 1 and begin line 2. This sub-element has two attributes; one to indicate the operator and one for the text itself. The maximum number of characters is 40. This reduces to 20 if used with printerTicket files.

```
<displayText  operator="" data="" />
```

5.4 printerFiscalReceipt

Emission of commercial documents (documenti commerciali) is performed in the same manner as for obsolete fiscal receipts (scontrini fiscali) in that the same printerFiscalReceipt XML file type is maintained.

These sub-elements can be divided into commands that are used to "manage" commercial documents and those that are used "inside" the individual print out which can consist of many records of different types. The former has only the operator attribute whereas the latter have many attributes that will be discussed further in detail.

Management commands:

N	Sub-element	Description
1	beginFiscalReceipt	Open commercial document
2	endFiscalReceipt	Close commercial document
3	printRecVoid	Cancel commercial document

Non-management commands:

N	Sub-element	Description
1	printRecItem	Item sale
2	printRecItemVoid	Cancel sale (Storno in Italy)
3	printRecItemAdjustment	Discount, surcharge or modifier
4	printRecItemAdjustmentVoid	Cancel discount, surcharge or modifier
5	printRecMessage	Additional row or description
6	printRecRefund	Refund (goods return or reso in Italy) * or modifier
7	printRecRefundVoid	Cancel refund or modifier
8	printRecRefundAdjustment	Alias of printRecRefund
9	printRecSubtotal	Print and/or display subtotal
10	printRecSubtotalAdjustment	Subtotal discount or surcharge
11	printRecSubtotalAdjustVoid	Cancel subtotal discount or surcharge
12	printRecTotal	Send payment
13	printGraphicCoupon	Print graphic coupon
14	printBarCode	Print barcode (or QR code)

* Refunds are automatically converted to cancellations (storni) in commercial sale documents. In commercial refund and/or void documents, printRecRefund requires SET 14/58 and/or 14/59 to be set accordingly to 1 / SI (not applicable to modifiers).

Note: Each commercial document file must comply with the following conditions:

- It must be opened
- It must include at least one sale/refund row
- The total amount cannot be negative
- The payment must be completed
- It must be closed

Multiple PrintRecTotal lines can be used if for example a customer pays partly by cheque and partly by cash (assuming the total due hasn't already been reached or exceeded).

Additional header lines must be placed before beginFiscalReceipt as they are preloaded in memory.

5.4.1 Commercial Document Response File

Responses contain the following additional information relative to the emitted commercial document:

- lastCommand – 74 (The printer status 1-074 command). See 3.11 Response Structure
- printerStatus – Five-byte status code. See Responses section for full explanation
- fiscalReceiptNumber – Daily sequence number without Z prefix
- fiscalReceiptAmount – Document value. Zero in the case of an automatic void
- fiscalReceiptDate – Document date
- fiscalReceiptTime – Document time
- zRepNumber – Z report sequence number for the day

An example response message:

```
<response success="true" code="" status="x">
  <addInfo>
    <elementList>lastCommand,printerStatus,fiscalReceiptNumber,fiscalReceiptAmount,fiscalReceiptDate,
fiscalReceiptTime,zRepNumber</elementList>
    <lastCommand>74</lastCommand>
    <printerStatus>20010</printerStatus>
    <fiscalReceiptNumber>nnnn</fiscalReceiptNumber>
    <fiscalReceiptAmount>yyyyy,cc</fiscalReceiptAmount>
    <fiscalReceiptDate>gg/mm/aaaa</fiscalReceiptDate>
    <fiscalReceiptTime>oo:mm</fiscalReceiptTime>
    <zRepNumber>zzzz</zRepNumber>
  </addInfo>
</response>
```

5.4.2 printRecItem – Sale Items

Prints sale items on a commercial sale document.
Prints refund items on a commercial refund document if flag SET 14/58 = 0 / NO.
Prints void items on a commercial void document if flag SET 14/59 = 0 / NO.

<printRecItem operator="" description="" quantity="" unitPrice="" department="" justification="" />

Optionally adding 50 to the operator number is interpreted by the printer so that the quantity line will be printed even if it's 1.

5.4.3 printRecItemVoid – Cancel Sale Items

Cancels the previous (last) sale operation or an earlier one. Cancelling an earlier (but not last) sale is known in Italy as a storno and can be the first transaction. By setting the attribute voidLastItem="1" the cancellation is deemed a correction rather than a storno. In this case, a printRecItem sub-element must immediately precede this sub-element and printRecItemVoid cannot be the first transaction. The absence of the voidLastItem attribute or if the value is "0", deems the operation a storno whereby specific sale information must be passed just as in a sale item operation.

A storno example:

<printRecItemVoid operator="" description="" quantity="" unitPrice="" department="" justification="" />

<printRecItemVoid voidLastItem="0" operator="" description="" quantity="" unitPrice="" department="" justification="" />

Optionally adding 50 to the operator number is interpreted by the printer so that the quantity line will be printed even if it's 1 (voidLastItem="0" only).

A void last item example (operator number offset must not be used):

<printRecItemVoid operator="" voidLastItem="1" />

5.4.4 printRecItemAdjustment

A discount, surcharge or modifier is applied. The discount or surcharge can be either with reference to the previous sale operation or to a specific department number. When one of the two "last sale" options are selected, the department value is ignored and can be null or the said attribute can be omitted. This is because the fiscal printer uses the department number relative to the last sale. When the last sale option is selected, it cannot be the first transaction in a commercial document or direct invoice. Modifiers require a specific department number. In all cases, the amount must be greater than zero.

```
<printRecItemAdjustment  operator="" adjustmentType="" description="" amount=""
department="" justification="" />
```

Where:

- **adjustmentType** determines discount/surcharge/modifier operation to perform:
  - 0 = Discount on last sale
  - 3 = Discount on a department
  - 5 = Surcharge on last sale
  - 8 = Surcharge on a department
  - 10 = Deposit (Acconto)
  - 11 = Omaggio (Free of Charge)
  - 12 = Buono monouso (single-use voucher)

Other values are not allowed. 10, 11 and 12 relate to modifiers.

operationType alias can also be used in place of adjustmentType.

Note: A zero amount will throw a printer error 16.

5.4.5 printRecItemAdjustmentVoid

Cancels the previous (last) discount, surcharge or modifier operation. A printRecItemAdjustment sub-element must immediately precede this sub-element.

```
<printRecItemAdjustmentVoid  operator="" />
```

5.4.6 printRecMessage

Applies additional header, additional sale descriptions or trailer lines to the commercial document. Also applies additional header and client lines to invoices (described in the printerFiscalDocument section) plus prints or erases the EFT-POS transaction lines and sets customer ID in JSON file. Can also be used in management documents (described in the printerNonFiscal section). One command is required for each printed row except for the EFT-POS transaction lines whereby this sub-element instructs the fiscal printer to print or erase all EFT-POS transaction lines from memory. Additional header lines must be sent prior to beginFiscalReceipt or prior to the commercial refund or void special opening command. The command can also be used in direct invoices with messageType 4.

```
<printRecMessage  operator="" messageType="" index="" font="" message="" /> or
<printRecMessage  operator="" messageType="4" message="" or
<printRecMessage  operator="" messageType="8" clearEFTPOSBuffer="" />
```

where:

- **messageType** defines the row type to be printed:
  - 1 = Additional header. This type must be placed before the beginFiscalReceipt sub-element
  - 2 = Trailer (after NUMERO CONFEZIONI and before NUMERO CASSA)
  - 3 = Additional trailer (promo lines after NUMERO CASSA and before barcode or QR code)
  - 4 = Additional description (in the body of the commercial document or direct invoice)
  - 7 = Customer Id. Sets CustomerId field in www/json_files/rec.json file (see Fiscal Printer Intelligent Features Guide for details). The font has no relevance so the attribute can be omitted
  - 8 = Print or erase all EFT-POS transaction lines

- **index** indicates the line number:
  - Range 1 to 9 for additional header (type 1)
  - Range 1 to 99 for trailer and additional trailer descriptions (types 2 and 3)
  - No meaning for additional row, Customer Id and EFT-POS transaction lines (types 4, 7 and 8)
  - The attribute can be omitted

- **font** attribute can be omitted when messageType is either 4, 7 or 8

- **message** represents the text to be printed or the customer ID. The maximum lengths are as follows:
  - Message type 4 = Max 38 (or 37 with invoices)
  - Message type 7 = Max 46 (although native protocol limit is 64)
  - Message type 8 = Not applicable. Attribute can be omitted
  - All other message types = Max 46

  Further characters are truncated.

- **clearEFTPOSBuffer** attribute is only relevant when messageType is 8:
  - 0 = Print EFT-POS transaction lines
  - 1 = Cancel EFT-POS transaction lines

5.4.7 printRecRefund

Prints refund items on a commercial refund document if flag SET 14/58 = 1.
Prints void items on a commercial void document if flag SET 14/59 = 1.
Converts automatically to a correction (storno) on a commercial sale document.
Can also be used with modifiers.

When goods are returned by a customer, a commercial refund document must be emitted. This is equivalent to "RESO MERCE" in Italy. See the 5.5 Commercial Refund Document (Documento di Reso) section.

```
<printRecRefund  operator="" description="" quantity="" unitPrice="" department="" justification=""
/>
```

Optionally adding 50 to the operator number is interpreted by the printer so that the quantity line will be printed even if it's 1.

When used with modifiers:

```
<printRecRefund  operator="" operationType=""  description="" amount="" department=""
justification="" />
```

Where:

- **operationType** determines modifier operation to perform:
  - 10 = Deposit (Acconto)
  - 11 = Omaggio (Free of Charge)
  - 12 = Buono monouso (single-use voucher)

Other values are not allowed.

adjustmentType alias can also be used in place of operationType.

Note: A zero amount will throw a printer error 16.

5.4.8 printRecRefundAdjustment

Alias of printRecRefund.

5.4.9 printRecRefundVoid

Cancels the previous (last) refund or modifier operation. A printRecRefund sub-element must immediately precede this sub-element.

```
<printRecRefundVoid  operator="" />
```

5.4.10 printRecLotteryID

This command pre-loads the national lottery unique customer code and should be placed before printRecTotal.

```
<printRecLotteryID  operator="" code="" />
```

where:

- **code** specifies the unique code

5.4.11 printRecSubtotal

This command allows the real-time subtotal to be printed and/or shown on the display.

```
<printRecSubtotal  operator="" option="" />
```

where:

- **option** sets the subtotal option:
  - 0 = Print and show on the display
  - 1 = Only print
  - 2 = Only show on the display

  If option 3 (request subtotal amount) is passed, the FpMate CGI service ignores this sub-element and continues to the next one.

5.4.12 printRecSubtotalAdjustment

A discount or surcharge is applied on the subtotal. The department attribute is ignored and can be null or the attribute itself can be omitted. Discounts and surcharges are applied automatically and proportionally to each department based on the individual department total for this specific commercial document or direct invoice. Amount must be greater than zero. Any departments programmed with the "Sales attribute" set to 01 are ignored (from BN 252). If a modifier transaction precedes this command, firmware BN 252 or greater is required otherwise Error 17 is returned.

```
<printRecSubtotalAdjustment  operator="" adjustmentType="" description="" amount=""
justification="" />
```

where:

- **adjustmentType** determines discount/surcharge operation to perform:
  - 1 = Discount on subtotal with subtotal printed out
  - 2 = Discount on subtotal without subtotal printed out
  - 6 = Surcharge on subtotal with subtotal printed out
  - 7 = Surcharge on subtotal without subtotal printed out

Other values are not allowed.

Note: A zero amount will throw a printer error 16.

5.4.13 printRecSubtotalAdjustVoid

Cancels the previous (last) discount/surcharge on subtotal operation. Fixed wording is printed so only the operator attribute needs to be passed. A printRecSubtotalAdjustment sub-element must immediately precede this sub-element.

```
<printRecSubtotalAdjustVoid  operator="" />
```

5.4.14 printRecTotal

One or more commands can be sent; more than one means that the payment is composed of several partial payments. In this case, once the total has been reached or exceeded, no more payment commands can be sent.

Note that insufficient payments are reflected in the wording "DIFFERENZA" on the display whereas payments with change show "RESTO". If the payment amount exactly equals the total outstanding to pay, the description attribute value will be shown on the display with the appropriate justification. The description on the document however will always be the description attribute value.

```
<printRecTotal  operator="" description="" payment="" paymentType="" index="" justification=""
/>
```

where:

- **paymentType** indicates the payment method:
  - 0 = Cash
  - 1 = Cheque
  - 2 = Credit or credit card. Credit now interpreted as mixed not paid
  - 3 = Ticket
  - 4 = Multiple tickets
  - 5 = Not paid
  - 6 = Payment discount

- **index** can be used with cash, credit cards or tickets to select a specific totaliser. In the case of multiple tickets, index is used to indicate the quantity. With respect to Not paid and Payment discount, index specifies the sub-type. index has no relevance with cheques. The following ranges are available:
  - Cash = 0 to 5
  - Credit = 0 (same as paymentType 5 index 0)
  - Credit card = 1 to 10
  - Ticket = 1 to 10
  - Multiple tickets = 1 to 99
  - Not paid is as follows:
    - 0 = Mixed (goods and services)
    - 1 = Goods
    - 2 = Services
    - 3 = Invoice to follow
    - 4 = RT invoice (for future use)
    - 5 = SSN (National Health Service)
  - Payment discount is as follows:
    - 0 = Generic
    - 1 = Multi-use voucher

The Communication Protocol document describes in detail the 1-084 command.

5.4.15 printGraphicCoupon

One graphic coupon can be printed at the end of the commercial document after the additional row, description or barcode. The sub-element must be placed between beginFiscalReceipt and endFiscalReceipt even though the coupon is printed after all the trailer lines, barcode / QR code and FOOTER. The binary data must be supplied in the base64 format and is passed as element data after the attributes rather than being a specific attribute value. The sub-element cannot be auto closed with /> and must therefore contain the closing sub-element name prepended with /. Do not use this command to print coupons without base64 data – It is not meant to be used with files uploaded via the upload.cgi service. The maximum Base 64 data size is 26668 bytes which corresponds with an unencoded limit of 20000 bytes.

<printGraphicCoupon operator="" graphicFormat="" >*base64 data*</printGraphicCoupon>

where:

- **graphicFormat** indicates the original image format (irrespective of the base 64 encoding):
    - B = BMP
    - R = Raster (file created with the Epson BMP to Raster utility or ESC/POS sequence)

The data attribute is obsolete and should be omitted.

5.4.16 printBarCode

Barcodes or QR codes are printed at the end of the commercial document after the additional trailer lines but before the FOOTER. Only one barcode can be printed in a commercial document unless the paper cut native command is used (1-137). Different attributes can be used depending on the barcode type:

For standard barcodes:

<printBarCode operator="" position="" width="" height="" hRIPosition="" hRIFont="" codeType="" code="" />

For QR codes:

<printBarCode operator="" qRCodeAlignment="" qRCodeSize ="" qRCodeErrorCorrection="" codeType="" code="" />

where:

- **position** defines the starting position from the left margin (range 0 to 511). Three special values can also be used:
    - 900 = Left aligned
    - 901 = Centred
    - 902 = Right aligned

  This attribute is not used with QR codes and can therefore be omitted.

- **qRCodeAlignment** defines the QR code position as follows:
    - 0 = Left aligned
    - 1 = Centred
    - 2 = Right aligned

- **width** indicates the print dot width of each distinct bar (range 1 to 6). This attribute is not used with QR codes and can therefore be omitted. Please note that not all readers are able to read barcodes with a 1 dot width

- **height** indicates the height of the bar code measured in print dots (range 1 to 255). This attribute is not used with QR codes and can therefore be omitted

- **qRCodeSize** indicates the QR code dimension (range 1 to 16)

- **hRIPosition** selects one of three ways to print the alphanumeric representation of the barcode or to disable it altogether. This attribute is not used with QR codes and can therefore be omitted. The options are as follows:
    - 0 = Disabled
    - 1 = Above the barcode
    - 2 = Below the barcode
    - 3 = Below and above the barcode

- **qRCodeDataType** indicates whether the QR code data is alphanumeric or binary. When binary is chosen, the code attribute value is represented by pairs of hexadecimal digits. For example, HELLO = 48454C4C4F. The options are as follows:
    - alphaNumeric or 0
    - binary or 9

- **hRIFont** indicates the font to be used for the HRI string. This attribute is not used with QR codes and can therefore be omitted. The options are as follows:
    - A
    - B
    - C

- **qRCodeErrorCorrection** specifies the level of error correction to employ (range 0 to 3):
    - 0 = Low (smallest dimension)
    - 1 = Medium low
    - 2 = Medium high
    - 3 = High (largest dimension)

- **codeType** indicates the barcode or QR code standard. Choose from one of the following:

  - UPC-A / 65
  - UPC-E / 66
  - EAN13 / 67
  - EAN8 / 68
  - CODE39 / 69
  - ITF / 70
  - CODABAR / 71
  - CODE93 / 72
  - CODE128 / 73
  - 74 to 78 *
  - QRCODE1 / 91
  - QRCODE2 / 92

* 74 to 78 do not have an alphanumeric alias. See Communication Protocol document for codes types (command H1=1 H2=075).

- **code** represents the barcode or QR code itself. QR codes up to 256 characters can be printed. If the qRCodeDataType attribute indicates a binary representation, this attribute value is represented by pairs of hexadecimal digits. For example, HELLO = 48454C4C4F

Important Notes:

- The character set and the string length are defined by the standards related to each bar code system

- The starting position, the string length and the barcode width must be defined according to the paper roll width. If there is not enough room width wise, the barcode will not be printed at all

- According to the CODE128 standard, one of the following character sets must be specified:

  - CODE A. To select CODE A, the first two characters of the "code" attribute must be "{A"
  - CODE B. To select CODE B, the first two characters of the "code" attribute must be "{B"
  - CODE C. To select CODE C, the first two characters of the "code" attribute must be "{C"

  When employing CODE128, note that these first two characters are not part of the bar code itself and are therefore not printed. They are only present to select the character set.

- The four attributes qRCodeAlignment, qRCodeSize, qRCodeErrorCorrection and qRCodeDataType should only be used if codeType is QRCODE1 or QRCODE2

- The five attributes position, width, height, hRIPosition and hRIFont should <u>not</u> be used if codeType is QRCODE1 or QRCODE2

- CODABAR strings must contain the start and stop characters.

- QRCODE2 is the preferred variant. QRCODE1 format is often incompatible with Smartphone apps

Please refer to chapter entitled H1=1; H2=075 – PRINT BARCODE OR QR CODE in the Communication Protocol document for complete details.

5.5 Commercial Refund Document (Documento di Reso)

Epson RT fiscal printers can emit commercial refund documents related to goods refund operations. The commands allowed depend on the SET 14/58 flag. In Ndc mode (1 / SI), the same commands as for obsolete credit notes are allowed and the same rules are followed. In non-Ndc mode (0 / NO), the same commands as for commercial documents are allowed and the same rules are followed. The root element remains printerFiscalReceipt. If the original document exists on the EJ, individual totals for each of the 19 tax groups present in the refund document cannot exceed the same totals in the original document. To check whether a specific document can be used as the reference document, a query document status request can be performed (9-205 command).

5.5.1 Commercial Refund Document Response File

Commercial refund document responses are identical to commercial document responses.

5.5.2 Opening a Commercial Refund Document

To open a commercial refund document, a special type 4 "printRecMessage" sub-element must be used that has to conform to one of two possible formats:

- REFUND zzzz nnnn ddmmyyyy sssssssssss

where:

- **zzzz** equals the four-digit Z report number
- **nnnn** equals the four-digit document number
- **ddmmyyyy** equals the date
- **sssssssssss** equals the eleven-character fiscal serial number of the printer. Can also be one of the three POS, VR and ND keywords

It is important to use capitals and separate the fields with blanks. beginFiscalReceipt is not required with commercial refund documents but can be still used for conformity if desired (placing it after printRecMessage). If the fiscal serial number is not the same as the printer that receives the request, an EJ search is not performed. Currently there is no fiscal printer intercommunication function. The special printRecMessage command must be the first in the commercial refund document sequence (but after any type 1 "printRecMessage" optional additional header lines). The RESO MERCE wording has been deprecated. Please see the Communication Protocol document for full details.

5.5.3 Closing a Commercial Refund Document

The same rules as for commercial documents are followed.

5.6 Commercial Void Document (Documento di Annullo)

Epson RT fiscal printers can print out commercial void documents. It means for example that an erroneously printed commercial document can be completely voided. With automatic printing, only printRecMessage is used otherwise the commands allowed depend on the SET 14/59 flag. In Ndc mode (1 / SI), the same commands as for obsolete credit notes are allowed and the same rules are followed. In non-Ndc mode (0 / NO), the same commands as for commercial documents are allowed and the same rules are followed. printRecMessage can be followed by the original transactions in case the document is not found. If the original document exists on the EJ, the opening command will automatically print and close the new void document. In this case, any further commands are ignored. Before proceeding, a query document status request can be optionally performed to ascertain whether a specific document can be used as the reference document (9-205 command).

5.6.1 Commercial Void Document Response File

Automatic printRecMessage responses and non-automatic responses are identical to commercial document responses.

5.6.2 Opening or Automatically Printing a Commercial Void Document

To open or automatically print a commercial void document, a special type 4 "printRecMessage" sub-element must be used that has to conform to one of two possible formats:

- VOID zzzz nnnn ddmmyyyy sssssssssss

where:

- **zzzz** equals the four-digit Z report number
- **nnnn** equals the four-digit document number
- **ddmmyyyy** equals the date
- **sssssssssss** equals the eleven-character fiscal serial number of the printer

It is important to use capitals and separate the fields with blanks. beginFiscalReceipt is not required with commercial void documents but can be still used for conformity if desired (placing it after printRecMessage). If the fiscal serial number is not the same as the printer that receives the request, an EJ search is not performed. Currently there is no fiscal printer intercommunication function. The special printRecMessage command must be the first in the commercial void document sequence (but after any type 1 "printRecMessage" optional additional header lines). The ANNULLAMENTO wording has been deprecated. If the document is found and a commercial void document is automatically printed, XML lines after printRecMessage are not processed. If the document was found but flag SET 14/68 ANNULLO SEMI MAN. = 1 / SI, the void document remains open. In this case, the replicated document must be fiscally identical (partial void is not permitted). Please see the Communication Protocol document for full details.

5.6.3 Closing a Commercial Void Document (non-automatic only)

The same rules as for commercial documents are followed.

5.7 printerNonFiscal

Emission of management documents (documenti gestionali) is performed in the same manner as for obsolete non-fiscal receipts (scontrini non-fiscali) in that the same printerNonFiscal XML file type is maintained.

The relative sub-elements can be divided into commands that are used to "manage" them and those that are used "inside" the individual document (which consists of only a few commands when compared to a commercial document). Management sub-elements only have the operator attribute whereas the others have many attributes that will be discussed further in detail. printerNonFiscal cannot be used with non-fiscal TM printers.

Management commands:

N	Sub-element	Description
1	beginNonFiscal	Open management document
2	endNonFiscal	Close management document

Non-management commands:

N	Sub-element	Description
1	printBarCode	Print barcode or QR code
2	printGraphicCoupon	Print graphic coupon
3	printNormal	Print row
4	printRecMessage	Prints or erases EFT-POS response lines
5	setLogo	Uploads logos to NVRAM and programs SET15 HEADER and FOOTER indexes

5.7.1 Management Document Response File

Responses only contain the two basic additional information entries – lastCommand (see 3.11 Response Structure) and printerStatus. An example response message:

```
<response success="true" code="" status="x">
        <addInfo>
                <elementList>lastCommand,printerStatus</elementList>
                <lastCommand>74</lastCommand>
                <printerStatus>00100</printerStatus>
        </addInfo>
</response>
```

5.7.2 printBarCode

The printing of barcodes and QR codes in management documents uses the same commands and rules as described in the printerFiscalReceipt section 5.4.16 printBarCode. However, unlike a commercial document (in standard mode without paper cut deactivation), there is no limit to the number of barcodes or QR codes that can be printed. Barcodes and QR codes are printed immediately.

5.7.3 printGraphicCoupon

The printing of graphic coupons in management documents uses the same commands and rules as described in the printerFiscalReceipt section 5.4.15 printGraphicCoupon. The only difference is that while in commercial document mode the graphic coupon can only be printed once at the end of the print-out, in management document mode it can be printed anywhere in the document body and more than once. The printing of several different graphic images is therefore allowed. The sub-element must be used between beginNonFiscal and endNonFiscal.

The data attribute is obsolete and should be omitted.

5.7.4 printNormal

Prints a row. All four fonts are supported. The format is as follows:

```
<printNormal  operator="" font="" data="" />
```

where:

- **data** contains the text to be printed (max 46 characters)

5.7.5 printRecMessage

When used in a management document, the **messageType** attribute must be set to 8 (Print or Erase all EFT-POS transaction lines). The only attribute necessary apart from the operator is **clearEFTPOSBuffer.** All other attributes can be omitted. See 5.4.6 printRecMessage.

5.7.6 setLogo

setLogo in a management document can perform one of three functions:

1. Logo upload to NVRAM or
2. Index programming (SET 15 / 09, 10, 19 and 20) or
3. Both the above

When uploading logos to NVRAM, the binary data must be supplied in the base64 format. It is passed as element data after the attributes rather than being a specific attribute value. In this case, the sub-element cannot be auto closed with /> and must therefore contain the closing sub-element name prepended with /. Graphic logo uploading to NVRAM must be done via a management document. setLogo can also be used in a printerCommand or printerCommands file but in this case only the indexes can be programmed. Do not use this command to upload logos to NVRAM without base64 data – It is not meant to be used with files uploaded via the upload.cgi service. The maximum Base 64 data size is 26668 bytes which corresponds with an unencoded limit of 20000 bytes.

```
<setLogo  operator="" location="" index="" option="0" graphicFormat="" >base64 data</setLogo>
<setLogo  operator="" index="1" option="1" graphicFormat="B" >base64 data</setLogo>
<setLogo  operator="" option="1" graphicFormat="R" >base64 data</setLogo>
<setLogo  location="" index="" option="2" />
```

where:

- **location** defines where the logo will be printed. To program two different logos as HEADER and FOOTER would require two distinct setLogo sub-elements. They could of course reside in a single XML file. The options are as follows:
  - 0 = All positions (HEADER and FOOTER for commercial documents, management documents plus HEADER and FOOTER for invoices)
  - 1 = Commercial document and management document HEADER only
  - 2 = Commercial document and management document FOOTER only
  - 3 = Invoice HEADER only
  - 4 = Invoice FOOTER only

- **index** specifies the number that will be programmed in the positions specified by the location attribute. Values range from 0 to 9. Zero deactivates logo printing. index is also required if the file format is BMP, but in this case must not be zero. Raster files created by the BMP to Raster utility on the other hand already contain the index in the .bin file header (based on the key code pair values specified during the conversion)

- **option** specifies one of three modes as follows:
  - 0 = Upload and SET 15 programming
  - 1 = Upload only
    In this case, the location attribute is not necessary, the index attribute is only necessary if the graphic format is BMP
  - 2 = SET 15 programming only
    In this case since base64 data is not supplied, the setLogo sub-element can be auto closed with /> and the graphicFormat attribute can be omitted. The operator attribute is also not needed

- **graphicFormat** specifies one of two formats (before base 64 encoding) as follows:
    - B = BMP
    - R = Raster (File created with Epson BMP to Raster utility)

The data attribute is obsolete and should be omitted.

5.8 printerFiscalReport

The FpMate CGI service can instruct the printer to perform a daily fiscal closure (Z report), a daily financial report (X report) or both. Only the operator attribute is necessary. A timeout attribute can be used (except printXReport). This is necessary to allow time for data transmission to the tax authority server. Only one report command is used per request file. Common commands such as displayText can however be used but must be placed before the report command. See examples at the end of this document.

Sub-element	Description
printXReport	Daily financial report (X Report). Default timeout six seconds
printZReport	Daily fiscal closure (Z Report). * Default timeout six seconds
printXZReport	X Report followed immediately by Z report. * Default timeout 12 seconds

* Data transmission to the tax authority server occurs automatically.

5.8.1 Report Response File

Report responses contain the following additional information:

- lastCommand – See 3.11 Response Structure
- printerStatus – Five-byte status code. See Responses section for full explanation
- zRepNumber – Equals the Z report sequence number (the emitted closure)
- dailyAmount – Equals the cumulative daily amount

An example response message:

```
<response success="true" code="" status="x">
        <addInfo>
                <elementList>lastCommand,printerStatus,zRepNumber,dailyAmount</elementList>
                <lastCommand>74</lastCommand>
                <printerStatus>20110</printerStatus>
                <zRepNumber >11</ zRepNumber>
                <dailyAmount >332,33</ dailyAmount>
        </addInfo>
</response>
```

5.8.2 Report Request Types

The following three report types are supported:

N	Sub-element	Description	XML
1	printXReport	Financial report only	<printXReport operator="" />
2	printZReport	Fiscal daily closure only	<printZReport operator="" timeout="" />
3	printXZReport	Financial report and fiscal daily closure (in that order)	<printXZReport operator="" timeout="" />

The "timeout" value is expressed in milliseconds.

5.9 printerCommand

The FpMate CGI service has the capability to extract administrative/diagnostic information from a fiscal printer as well as perform resets. Only one command can be used per request file. If you wish to send more than one, please use the alternative printerCommands keyword. Common commands such as displayText but not directIO can however be used but must be placed before the main command. See examples at the end of this document. The following table lists the commands:

N	Sub-element	Description
1	authorizeSales	Activates EFT-POS electronic payment
2	beginTraining	Activates training mode (MF mode only). If commercial documents have been issued following a fiscal closure, this command is inhibited (a closure must be requested beforehand either by executing a printXZReport or a printZReport command)
3	EFTPOSDailyClosure	Instructs POS terminal to perform its own daily closure
4	EFTPOSGetCurrentTotal	Reads current daily totals from POS terminal
5	endTraining	Deactivates training mode (MF mode only)
6	getDate	Retrieves the printer date and time
7	printContentByDate	Prints Electronic Journal data.
8	printContentByNumbers	
9	printDuplicateReceipt	Prints last emitted commercial document. Must be logged in
10	printRecCash	Performs CASH IN or CASH OUT operations
11	printRecVoid	Cancels any open commercial document or direct invoice
12	queryContentByDate	Reads Electronic Journal data. Must be logged in
13	queryContentByNumbers	
14	queryPrinterStatus	Retrieves printer basic or RT status
15	resetPrinter	Resets fiscal printer. Cancels any open commercial documents or invoices. Closes any open management documents
16	setDate	Sets the date and time
17	setLogo	Programs SET 15 HEADER and FOOTER indexes for commercial documents, management documents and invoices

5.9.1 Command Response File

Command responses incorporate the basic additional information entry – lastCommand (see 3.11 Response Structure) with command specific elements which are described in each separate command.

A typical response example:

```
<response success="true" code="" status="x">
        <addInfo>
                <elementList>lastCommand,printerStatus</elementList>
                <lastCommand>74</lastCommand>
                <cpuRel>07.00</cpuRel>
                <mfRel>04.3</mfRel>
                <mfStatus>0</mfStatus>
                <fpStatus>00110</fpStatus>
        </addInfo>
</response>
```

5.9.2 authorizeSales

Activates EFT-POS payment authorisation. The only attributes necessary are operator and amount.

```
<authorizeSales  operator="" amount="" />
```

The following additional information is provided in the response:

```
        <elementList>lastCommand,printerStatus,paymentResult,lineCount</elementList>
        <lastCommand>278</lastCommand>
        <printerStatus>00110</printerStatus>
        <paymentResult><Max 122-byte string from POS></paymentResult>
        <lineCount>31
                <lineNumber1>  INGENICO ITALIA SPA                </lineNumber1>
                <lineNumber2>   DIVISIONE MONETICA                </lineNumber2>
                        …
                        …
                <lineNumber31>Nessun addebito eseguito            </lineNumber31>
        </lineCount>
```

5.9.3 beginTraining (MF mode only)

Does not currently enter "Demo RT" mode.

No attributes are required. Entering training mode is only possible if no commercial documents have been issued following a daily fiscal closure.

```
<beginTraining />
```

5.9.4 EFTPOSDailyClosure

Instructs POS terminal to perform its own daily closure.

```
<EFTPOSDailyClosure  operator="" terminalID="" registerID="" />
```

where:

- **terminalID** should correspond with the SET 31 POS terminal ID setting
- **registerID** usually set to 00000000 (Fiscal printer ID field normally ignored by POS terminals)

Depending on the outcome, the following additional information is provided in the response:

Positive Outcome (transactionResult = 00)

```
<elementList>lastCommand,terminalID,messageCode,transactionResult,terminalDailyTotal,hostDa
ilyTotal,EFTPOSReplyString</elementList>
<lastCommand>139</lastCommand>
<terminalID></terminalID>
<messageCode>C</messageCode>
<transactionResult>00<transactionResult>
<terminalDailyTotal></terminalDailyTotal>
<hostDailyTotal></hostDailyTotal>
<EFTPOSReplyString><Entire POS terminal reply string></EFTPOSReplyString>
```

Both totals are expressed as a 16-digit amount in cents without a decimal separator and with leading zeroes.

Example EFTPOSReplyString:

- 123456780C00000000000000002600000000000000026

Negative Outcome (transactionResult = 01)

```
<elementList>lastCommand,terminalID,messageCode,transactionResult,failureDescription,actionC
ode,EFTPOSReplyString</elementList>
<lastCommand>139</lastCommand>
<terminalID></terminalID>
<messageCode>C</messageCode>
<transactionResult>01<transactionResult>
<failureDescription></failureDescription>
<actionCode></actionCode>
<EFTPOSReplyString><Entire POS terminal reply string></EFTPOSReplyString>
```

failureDescription contains the eventual error description generated by the POS terminal of up to 19 characters.

actionCode contains the eventual code generated by the POS terminal of up to three bytes.

Example EFTPOSReplyString:

- 123456780C01ERRORE CHIUSURA 1239990000000000

5.9.5 EFTPOSGetCurrentTotal

Request current daily totals from POS terminal.

<EFTPOSGetCurrentTotal operator="" terminalID="" registerID="" />

where:

- **terminalID** should correspond with the SET 31 POS terminal ID setting
- **registerID** usually set to 00000000 (Fiscal printer ID field normally ignored by POS terminals)

The following additional information is provided in the response:

```
<elementList>lastCommand,terminalID,messageCode,transactionResult,currentTotal,EFTPOSRepl
yString</elementList>
<lastCommand>139</lastCommand>
<terminalID></terminalID>
<messageCode>T</messageCode>
<transactionResult><00 = positive or 01 = negative><transactionResult>
<currentTotal></currentTotal>
<EFTPOSReplyString><Entire POS terminal reply string></EFTPOSReplyString>
```

Total is expressed as a 16-digit amount in cents without a decimal separator and with leading zeroes.

Example EFTPOSReplyString:

- 123456780T000000000000000026000000

5.9.6 endTraining (MF mode only)

Does not currently exit "Demo RT" mode.

No attributes are required.

<endTraining />

5.9.7 getDate

No attributes are required. Retrieves the fiscal printer current date and time.

<getDate />

The following additional information is provided in the response:

```
<elementList>day,month,year,hour,minute</elementList>
<day></day>
<month></month>
<year></year>
<hour></hour>
<minute></minute>
```

5.9.8 printContentByDate

The printer must be in the logged in condition otherwise Error 17 impossible now is returned.

Used to reprint commercial documents, invoices and ticket box office slips that have been stored in the MPD (EJ).

<printContentByDate operator="" dataType="" fromDay="" fromMonth="" fromYear="" toDay="" toMonth="" toYear="" />

where:

- **dataType** indicates the type of data to collect:
  - 0 = All
  - 1 = Commercial documents (including refunds and voids)
  - 2 = Invoices
  - 3 = Box office tickets (Titoli di Accesso)
  - 4 = Obsolete
  - 5 = Obsolete
- **fromDay** indicates the start day expressed in one or two digits
- **fromMonth** indicates the start month expressed in one or two digits
- **fromYear** indicates the start year expressed in either two or four digits
- **toDay, toMonth and toYear** together indicate the end date in the same format as the three "from" attributes

5.9.9 printContentByNumbers

The printer must be in the logged in condition otherwise Error 17 impossible now is returned.

<printContentByNumbers operator="" dataType="" day="" month="" year="" fromNumber="" toNumber="" />

where:

- **dataType** has the same options as printContentByDate
- **day** indicates the day expressed in one or two digits
- **month** indicates the month expressed in one or two digits
- **year** indicates the year expressed in either two or four digits
- **fromNumber** indicates the first number of the commercial document to print for example. Values range from 1 to 9999
- **toNumber** indicates the last number of the commercial document to print for example. Values range from 1 to 9999

5.9.10 printDuplicateReceipt

The printer must be in the logged in condition otherwise Error 17 impossible now is returned.

The last commercial document (including refund or void variants) is reprinted. This command cannot print commercial documents issued prior to a fiscal daily closure. Since originals cannot by law be reprinted in the same format, copies are in fact management documents that contain the original lines as content read from the MPD (EJ).

```
<printDuplicateReceipt  operator="" />
```

5.9.11 printRecCash

This command is used when cash or cheques are paid in or out of the cash drawer.

```
<printRecCash  operator="" direction="" form="" amount="" />
```

where:

- **direction** is either "in" or "out"
- **form** is either "cash" or "cheque" / "check"

5.9.12 printRecVoid

The current commercial document or direct invoice is cancelled. endFiscalReceipt or endFiscalDocument is still required.

```
<printRecVoid  operator="" />
```

5.9.13 queryContentByDate

The printer must be in the logged in condition otherwise Error 17 impossible now is returned.

Used to collect and return EJ or MPD data in an array.

```
<queryContentByDate  operator="" dataType="" fromDay="" fromMonth="" fromYear="" toDay="" toMonth="" toYear="" />
```

Where the attributes follow the same rules as for printContentByDate.

The content is contained in the additional information portion in the following format:

```
<elementList>lastCommand,printerStatus,lineCount</elementList>
<lastCommand>74</lastCommand>
<printerStatus>00110</printerStatus>
<lineCount>
        <lineNumber1>xxxxxxxxxxxxxx</lineNumber1>
        <lineNumber2>xxxxxxxxxxxxxx</lineNumber2>
        And so on
</lineCount>
```

No fixed line limit has been set. The FpMate CGI service continually monitors the free memory available whilst the query is in progress. If no more free memory is available, the query is terminated. In this case, the last line wording is set to "OUT OF MEMORY" so that the user is notified.

5.9.14 queryContentByNumbers

The printer must be in the logged in condition otherwise Error 17 impossible now is returned.

Used to collect and return the Electronic Journal data in an array.

```
<queryContentByNumbers  operator="" dataType="" day="" month="" year="" fromNumber="" toNumber="" />
```

Where the attributes follow the same rules as for printContentByNumbers.

The content is contained in the additional information portion and is in the same format as queryContentByDate. The same limits apply.

5.9.15 queryPrinterStatus

Retrieves one of two types of fiscal printer status. The operator attribute is necessary plus a statusType indication attribute.

```
<queryPrinterStatus  operator="" statusType="" />
```

where:

- **statusType** is either "0" for basic status or "1" for RT status.

The following additional information is provided in the basic status response:

```
<elementList>lastCommand,cpuRel,mfRel,mfStatus,fpStatus</elementList>
<lastCommand>74</lastCommand>
<cpuRel></cpuRel>
<mfRel></mfRel>
<mfStatus></mfStatus>
<fpStatus></fpStatus>
```

where:

- **cpuRel** indicates the fiscal board firmware version
- **mfRel** indicates the MPR (fiscal memory) firmware version
- **mfStatus** indicates the MPR (fiscal memory) status (OK, FULL etc.)
- **fpStatus** indicates the printer status (same as printerStatus five-byte code). See Responses section for full explanation

The following additional information is provided in the RT status response:

```
<elementList>
lastCommand,rtType,rtMainStatus,rtSubStatus,rtDailyOpen,rtNoWorkingPeriod,rtFileToSend,
rtOldFileToSend,rtFileRejected,rtExpiryCD,rtExpiryCA,rtTrainingMode,rtUpgradeResult
</elementList>
<lastCommand>138</lastCommand>
<rtType></rtType>
<rtMainStatus></rtMainStatus>
<rtSubStatus></rtSubStatus>
<rtDailyOpen></rtDailyOpen>
<rtNoWorkingPeriod></rtNoWorkingPeriod
<rtFileToSend></rtFileToSend>
<rtOldFileToSend rtOldFileToSend>
<rtFileRejected></rtFileRejected>
<rtExpiryCD><rtExpiryCD>
<rtExpiryCA><rtExpiryCA>
<rtTrainingMode><rtTrainingMode>
<rtUpgradeResult><rtUpgradeResult>
```

where:

- **rtType** indicates the RT device type *
- **rtMainStatus** indicates the RT main status *
- **rtSubStatus** indicates the RT sub status *
- **rtDailyOpen** = indicates the logical DAY OPENED logical condition (0=closed and 1=open)
- **rtNoWorkingPeriod** indicates whether a Z report must be performed or not (0=no and 1=yes)
- **rtFileToSend** indicates the number of files due to be sent to the tax authority
- **rtOldFileToSend** indicates the number of files due to be sent to the tax authority but still waiting on the printer after a configurable number of days (SET 15/25)
- **rtFileRejected** indicates the number of files rejected by the tax authority
- **rtExpiryCD** indicates the device certificate expiry date in the yyyymmdd format
- **rtExpiryCA** = indicates the tax authority communication certificate expiry date in the yyyymmdd format
- **rtTrainingMode** = indicates the mode *
- **rtUpgradeResult** = indicates the last firmware update outcome *

* See Communication Protocol document.

5.9.16 rebootWebServer

No attributes are required. Reboots the printer's embedded web server. The response is:

```
<response success="true" code="" status="2">
```

No additional information is provided.

5.9.17 resetPrinter

The operator attribute is the only one required. This command will execute the following functions:

- Return the printer to the STATO REGISTRAZIONE state independent of the current state
- Close any open management documents
- Cancel any open commercial documents
- Cancel any open direct invoices printing two copies
- Cancel any open box office tickets plus return the printer to the "SC. FISC. DOC. COMM" mode
- Unblock the keyboard that was previously blocked with the 1-055 native protocol command
- Empty the keyboard buffer

```
<resetPrinter  operator="" />
```

5.9.18 setDate

Programs the fiscal printer date and time. It is only possible if no commercial documents have been issued following a daily fiscal closure (DAY OPENED = False).

```
<setDate  day="" month="" year="" hour="" minute="" />
```

where:

- **day** is either one or two digits. Leading zero is optional
- **month** is either one or two digits. Leading zero is optional
- **year** can be two or four digits
- **hour** is represented in the 24-hour format and is either one or two digits. Leading zero is optional
- **minute** is either one or two digits. Leading zero is optional

  It is not possible to set the number of seconds.

5.9.19 setLogo

This command is used to perform SET 15 logo programming and can specify where graphic logos are printed. When used in a printerCommand file, setLogo cannot be used to upload logos into NVRAM. Please refer to the setLogo section in the printerNonFiscal chapter for an explanation of the attributes (5.7.6 setLogo). Please note that only location and index are necessary and since base64 data is not supplied, the setLogo sub-element can be auto closed with />. The operator attribute is also not required.

```
<setLogo  location="" index="" />
```

The data attribute is obsolete and should be omitted.

5.10 printerCommands

This file type permits multiple elements. The command set is the same as for printerCommand except for the following elements as they generate specific responses:

- authorizeSales
- EFTPOSDailyClosure
- EFTPOSGetCurrentTotal
- getDate
- queryContentByDate
- queryContentByNumbers
- rebootWebServer

The response is the same as the queryPrinterStatus response.

5.11 printerFiscalDocument

Epson RT fiscal printers can also emit two types of invoice (fatture). These sub-elements can be divided into commands that are used to "manage" invoices and those that are used "inside" them. printerFiscalDocument cannot be used with non-fiscal TM printers. When thermal paper is used, two invoice copies are automatically printed. Modifiers are not supported.

Management commands:

N	Sub-element	Description
1	beginFiscalDocument	Open direct invoice or Request invoice based on last commercial document

2	endFiscalDocument or
	endFiscalReceipt
		Close invoice

Neither endFiscalDocument nor endFiscalReceipt are required with invoices based on the last commercial document.

Non-management commands:

Commands used with direct invoices:

N	Sub-element	Description
1	printRecItem	Item sale
2	printRecItemVoid	Cancel sale (Storno in Italy)
3	printRecItemAdjustment	Sale discount or surcharge. Modifiers are not supported
4	printRecItemAdjustmentVoid	Cancel sale discount
5	printRecMessage	Additional header line (type 5), client line (type 6) or additional description line (type 4) in the body of the direct invoice or trailer line (type 2)
6	printRecSubtotal	Subtotal cannot be printed on invoices. This command can only be used to display the subtotal (option = 2). Since it would normally be overwritten immediately, it is not usually used
7	printRecSubtotalAdjustment	Subtotal discount
8	printRecSubtotalAdjustVoid	Cancel subtotal discount
9	printRecTotal	Send payment
10	printBarCode	Print barcode or QR code (both copies)

Commands not used or supported with direct invoices:

N	Sub-element	Description
1	PrintRecRefund	Refunds (Goods return or reso in Italy are not supported in invoices. They are converted to corrections (storni). Modifiers are not supported in invoices
2	printRecRefundVoid	Cancel refund or modifier
3	printRecRefundAdjustment	Alias of printRecRefund
4	printGraphicCoupon	Print graphic coupon

Commands used with invoices based on the last commercial document:

N	Sub-element	Description
1	printRecMessage	Print additional header line or client line

5.11.1 Fiscal Document Response File

Fiscal document responses only contain the two basic additional information entries – lastCommand (see 3.11 Response Structure) and printerStatus). An example response message:

```
<response  success="true" code="" status="x">
        <addInfo>
                <elementList>lastCommand,printerStatus</elementList>
                <lastCommand>74</lastCommand>
                <printerStatus>00110</printerStatus>
        </addInfo>
</response>
```

5.11.2 Document Types

The following table illustrates the different document types and requests. Examples of each can be found in the examples section.

N	Sub type	documentType Attribute Identifier	Description
1	Based on the last commercial document	lastReceiptInvoice	Apart from the additional header and/or client lines, only beginFiscalDocument is sent. endFiscalDocument or endFiscalReceipt are unnecessary since the whole printout is automated
2	Direct	directInvoice	Any additional header and/or client lines must be sent first. Then after having opened a direct invoice, printerFiscalReceipt sub-elements are employed such as printRecItem, printRecTotal etc. rather than the obsolete printFiscalDocumentLine for example. Certain printerFiscalReceipt commands are not used or supported (see above). After payment, either endFiscalReceipt or endFiscalDocument can be sent

5.11.3 Direct Invoice Sub-Elements

The description limits are reduced from 38 to 37 characters. printRecTotal descriptions are not printed but are sent to the display so the character limit is reduced to 20 characters. printRecMessage is described below. printBarCode can be used to print a barcode or QR code (both copies).

5.11.4 beginFiscalDocument

The following XML line shows all the attributes for illustrative purposes:

```
<beginFiscalDocument  operator="" documentType="" documentNumber="" />
```

where:

- **documentType** can be one of the two types illustrated in the above table. Set the attribute based on the names shown in the appropriately named column of the above table

- **documentNumber** starts from 0 and runs up to 99999. Depending on the printer configuration, requests can be refused if numbers do not run in sequence. Use zero if you wish the fiscal printer to manage the invoice numbering. The printer automatically increments the invoice number after having printed it. The documentNumber attribute can also be omitted – the zero value will be assumed

- **documentAmount** is obsolete

5.11.5 printRecMessage

Additional invoice header and client lines must be sent prior to beginFiscalDocument. This sub-element can be used with both invoice types:

<printRecMessage operator="" messageType="" index="" message="" /> or
<printRecMessage operator="" messageType="4" message="" />

Where:

- **messageType** defines the row type to be printed:
  - 2 = Trailer lines (after TOTALE EURO line). Direct invoices only
  - 4 = Additional description (in the body of the Direct invoice)
  - 5 = Additional invoice header lines
  - 6 = Invoice client lines
- **index** indicates the line number:
  - Range 1 to 99 for trailer lines (type 2)
  - No meaning for additional row type 4. The attribute can be omitted
  - Range 1 to 20 for invoice additional header lines (type 5)
  - Range 1 to 5 for invoice client lines (type 6)
- **font** attribute is not necessary since the font is always normal
- **message** represents the text to be printed. The maximum lengths are as follows:
  - Message type 4 = Max 37
  - Message types 2, 5 and 6 = Max 46

  Further characters are truncated.

Regarding the additional invoice header lines, the SET 25 "RIGHE INIZIO FATTURA" reserved space setting on the fiscal printer must at least coincide with the index attribute setting. The default printer setting is zero which effectively disables the printing of type 5 lines even if the printer accepts the printRecMessage sub-element without returning an error.

5.11.6 printBarCode

Just as for commercial documents, only one barcode or QR code can be printed per invoice. The same code is printed on both copies at the foot of the print-out after an eventual graphic footer.

5.11.7 endFiscalDocument

This sub-element is used to close direct invoices.

```
<endFiscalDocument  operator=""  />
```

operationType is obsolete.

5.11.8 Important Prerequisites

- Previous set-up of the fiscal printer is required
- The option to print an invoice based on a commercial document is only possible if the request immediately follows a "printerFiscalReceipt" request. The invoice reports all the data present in the previous commercial document apart from barcodes and payment descriptions
- The invoice numbers must be **greater** than the last invoice number printed. But it doesn't have to be previous + 1, it could be previous + 5 for example. Alternatively, the value zero can be used so that it is managed by the printer
- Any additional headers and/or client lines must be sent before beginFiscalDocument

5.12 printerTicket

Epson RT fiscal printers can also emit box office tickets. The printer must be in shared mode whereby it can print both "titoli" and all the other types of print-out. Shared mode is deactivated by default. If you wish to print "titoli", please contact Epson since to enable shared mode requires printer intervention including the breaking of the fiscal seal (authorised technicians only). Box office tickets can also contain multiple barcodes plus it is also possible to use common commands such as openDrawer and displayText. Note that when printing box office tickets, text can only be shown on the top line of the display.

The relative sub-elements can be divided into commands that are used to "manage" box office tickets and those that are used "inside" the individual ticket which consists of only a few commands when compared to a commercial document. The operator attribute is not used. printerTicket cannot be used with non-fiscal TM printers.

Management commands:

N	Sub-element	Description
1	enterTicket	Switch to box office ticket mode
2	beginTicket	Open a new ticket
3	endTicket	Close ticket
4	exitTicket	Switch out of box office ticket mode to commercial document mode
5	printTicketVoid	Cancel ticket

Non-management commands:

N	Sub-element	Description
1	printTicket	Print one of two types of text line
2	printBarCode	Print barcode or QR code

5.12.1 Ticket Response File

Responses contain the following additional information:

- lastCommand See 3.11 Response Structure
- cpuRel
- mfRel
- mfStatus
- fpStatus

A basic response example:

```
<response success="true" code="" status="x">
        <addInfo>
                <elementList>lastCommand,printerStatus</elementList>
                <lastCommand>74</lastCommand>
                <cpuRel>07.00</cpuRel>
                <mfRel>04.3</mfRel>
                <mfStatus>0</mfStatus>
                <fpStatus>00110</fpStatus>
        </addInfo>
</response>
```

5.12.2 enterTicket

Switches into box office ticket mode.

```
<enterTicket  randomNum="" controlCode="" />
```

Where:

- **randomNum** equals a random number from 1 to 99
- **controlCode** equals the four-digit control code. After having signed a non-disclosure agreement (NDA), Epson can supply the instructions on how to calculate this code

5.12.3 beginTicket

Opens a new box office ticket.

```
<beginTicket  randomNum="" controlCode="" />
```

The attributes follow the same rules as for enterTicket.

5.12.4 printTicket

Prints one of two different types of text line in box office tickets. The two types are "normal" or "seal". Normal lines are printed and saved in the Electronic Journal. Seal lines of which there must be one per box office ticket are only printed. The printer returns an error if a second seal line sub-element is received. All four fonts are supported.

```
<printTicket  randomNum="" controlCode="" textType="R" font="1" text="" /> or
<printTicket  randomNum="" controlCode="" textType="S" font="1" text="" />
```

Where:

- **randomNum** and **controlCode** follow the same rules as for enterTicket
- **textType** equals one of two possible values:
  - R = normal text line
  - S = Seal text line of which there can only be one per box office ticket
- **Text** represents the text to be printed (up to 46 characters, further characters are truncated)

5.12.5 printTicketVoid

Cancels an open box office ticket. The printer remains in box office ticket mode.

```
<printTicketVoid  randomNum="" controlCode="" />
```

The attributes follow the same rules as for enterTicket.

5.12.6 printBarCode

The printing of barcodes and QR codes in box office tickets uses the same commands and rules as described in the printerFiscalReceipt section 5.4.16 printBarCode. There is no limit to the number of codes that can be printed in a box office ticket. They are printed immediately. The operator, randomNum and controlCode attributes are not required.

5.12.7 endTicket

Closes a box office ticket.

```
<endTicket  randomNum="" controlCode="" />
```

The attributes follow the same rules as for enterTicket.

5.12.8 exitTicket

Switches out of box office ticket mode to commercial document mode.

```
<exitTicket  randomNum="" controlCode="" />
```

The attributes follow the same rules as for enterTicket.

5.13 ePOS-Print (for TM printers only)

Please see ePOS documentation since the XML format is almost the same. The main difference is the name of the web service CGI file where the request is forwarded to:

• ePOS TM:	service.cgi
• ePOS fiscal FpMate:	fpmate.cgi

5.14 Aliases for Backward Compatibility

To aid the migration from the Windows EpsonFpMate version to the web service variant, the sub-element and attribute naming conventions can be maintained. Minor differences are explained. Note that directIO has a completely new format since the Active X / OCX software layer is not part of the fpmate.cgi code. The following tables illustrate the relative aliases:

Sub-elements:

N	FpMate CGI Service Name	Windows EpsonFpMate Alias
1	printRecVoid	printerVoidReceipt
2	printRecItemVoid	printRecVoidItem
3	printFiscalDocument	beginFiscalDocument (type 1)
4	printXZReport	printZTotReport
5	queryContentByDate	queryEjContent
6	queryContentByNumbers	queryEjContentByNumbers

Attributes:

N	FpMate CGI Service Name	Windows EpsonFpMate Alias
1	operator	Ope
2	message	Text (printRecMessage sub-element)
3	description	Text (Sale related sub-elements) PaymentDescription
4	code	Text (printBarCode sub-element)
5	data	Text (printNormal and displayText sub-elements)
6	documentLine	Text (printFiscalDocumentLine sub-element)
7	quantity	Qty
8	unitPrice	UnitCost
9	department	Dep
10	justification	Just
11	adjustmentType	Type
12	messageType	Type
13	paymentType	Type
14	mode	Type
15	option	Type
16	amount	Amount
17	index	Index Num
18	font	Font

N	FpMate CGI Service Name	Windows EpsonFpMate Alias

19	hRIFont	Font
		
		In addition, the following value aliases exist:
		
		A = FontA
		B = FontB
		C = FontC

20	Position	Pos
21	width	Width
22	height	Height
23	hRIPosition	Hri In addition, the following value aliases exist: 0 = DISABLED 1 = ABOVE 2 = BELOW 3 = TWICE
24	codeType	tipo
25	documentType	Document (1 = Invoice)
26	documentNumber	Number

6 Details of Fiscal ePOS-Print XML (Responses and Errors)

6.1 Description

The FpMate CGI service responses contain a minimum of three parameters and depending on the command may contain additional information. The first part of the response contains a <response element tag with the following three attributes:

N	Attribute	Description
1	success	true or false
2	code	String description. Usually empty; only filled during error conditions
3	status	Numeric identifier that follows the Epson ASB status codes as much as possible

If additional information is to be provided, the <addInfo> sub-element tag will be present. It has no attributes or data. It has as its first sub-element an <elementList> tag which contains as data a comma delimited list of the remaining element tag names. elementList has no attributes. These remaining elements contain data and generally have no attributes. Below is a successful commercial document response example (sent after having processed endFiscalReceipt):

```
<response success="true" code="" status="2">
  <addInfo>
    <elementList>lastCommand,printerStatus,fiscalReceiptNumber,fiscalReceiptAmount,fiscalReceiptDate,
fiscalReceiptTime</elementList>
    <lastCommand>74</lastCommand>
    <printerStatus>00010</printerStatus>
    <fiscalReceiptNumber>2</fiscalReceiptNumber>
    <fiscalReceiptAmount>1,00</fiscalReceiptAmount>
    <fiscalReceiptDate>26/01/2021</fiscalReceiptDate>
    <fiscalReceiptTime>12:13</fiscalReceiptTime>
  </addInfo>
</response>
```

The library presents the responses in an onreceive function containing three arrays and the XML:

N	Array name	Description
1	res	Contains success, code and status
2	tag_list_names	Contains the additional info parameter list
3	add_info	Contains the additional info parameters in a hash array. For example, use add_info.printerStatus to determine printer status
4	res_add	Contains all the XML

Please see the html/JavaScript example files for reference.

6.2 Success, Code and Status Cross Reference Table

Success	Code	Status	Comments
True	Always null	2 or TM ABS bitmap	Successful elaboration
False	NO_DATA	Always = 0	Empty HTTP POST SOAP request
	NO_RAM		Printer out of RAM memory
	PARSER_ERROR		Malformed XML file
	LAN_ERROR		FpMate CGI service fiscal firmware intercommunication timeout during initial communication phase (before XML line processing)
	LAN_TIME_OUT		FpMate CGI service fiscal firmware intercommunication timeout during XML line processing
	FP_NO_ANSWER		Fiscal firmware did not respond to FpMate CGI service command within timeout
	TM_NO_ANSWER/OFF_LINE		TM "Connection reset by peer" error or timeout
	CONFIGURATION_FILE_ERROR		Empty or non-existent secondary devices XML configuration file
	INCOMPLETE FILE		XML file does not contain the minimum elements necessary (missing endFiscalReceipt for example)
	Non valid XML command		Unknown or misplaced sub-element (endNonFiscal in an invoice for example)
False	EPTR_REC_EMPTY	Always 3	Problem due to cover open, printer offline or no paper roll loaded
False	PRINTER ERROR	Native fiscal protocol error code (except codes 2 and 3)	Problem due to fiscal printer at the native communication level. For example, discount amount = 0. For a complete list of possible errors, please refer to the "Communication Protocol" document and the "A.PDU Errors" chapter
False	EFT_POS_ERROR	Always = 38	Electronic payment error
False	FP_NO_ANSWER_NETWORK	Always = 0	JavaScript library ontimeout event or HTTP error. For example: HTTP 404 – No page found or network error

Success	Code	Status	Comments
False	incomplete XML commands	Always = 0	Deprecated
False	TM_OFF_LINE	0x80000001	TM printer out of paper for example.
	GENERIC		For future use

6.3 printerStatus and fpStatus

The fiscalprint.js JavaScript library contains a commented out decodeFpStatus function that could be used to decode the five-byte printerStatus / fpStatus codes into a human readable format.

The "Communication Protocol" document also contains a table that indicates the meaning of each possible value. It can be found under the section entitled "H1=1; H2=074 – GET PRINTER STATUS".

6.4 Parsing Errors

The FpMate CGI service performs XML validation checks and will respond immediately if a parser error is detected. Below is an example response:

```
<response success="false" code="PARSER ERROR" status="0">
</response>
```

6.5 Command Errors

The FpMate CGI service performs command validation checks and will respond immediately if a command error is detected. A typical error could be an illegal quantity or amount. Below is an example response when quantity is zero:

```
<response success="false" code="PRINTER ERROR" status="21">
        <addInfo>
                <elementList>lastCommand,printerStatus</elementList>
                <lastCommand>80</lastCommand>
                <printerStatus>00100</printerStatus>
        </addInfo>
</response>
```

If a sub-element name is misspelt or a sub-element relevant in one type of file is placed in a different type of file where it is not supported, the non-valid XML command response is returned (without additional information and status 0):

```
<response success="false" code="non valid XML command" status="0">
</response>
```

6.6 Incomplete or empty files

The FpMate CGI service returns an "INCOMPLETE FILE" response message if endFiscalReceipt is missing for example:

```
<response success="false" code=" INCOMPLETE FILE" status="0">
</response>
```

6.7 Secondary Printer Errors

In addition to the above errors, if the printer (where fpmate.cgi resides) is unable to communicate with the secondary printer, an appropriate error is returned. Below is an example response:

```
<response success="false" code="LAN ERROR" status="0">
</response>
```

6.8 FpMate CGI Service Timeout

The fiscal firmware did not respond to the FpMate CGI service command within the timeout period. Could be due to a long print-out such as bulk reprinting.

```
<response success="false" code="FP_NO_ANSWER" status="0">
</response>
```

This error can be simulated by activating the XON-XOFF protocol on the fiscal printer (SET 14/35).

6.9 XMLHTTP Request Timeouts

If no response arrives from the FpMate CGI service, an onerror function is called. Each browser responds in its own way with its own timeout setting and xhr.status value. The table below illustrates some major browsers behaviour.

N	Browser	Timeout	xhr.status
1	Internet Explorer 8	40 seconds	12029
2	Mozilla Firefox	15 seconds	0
3	iPad Safari	75 seconds	0

Software developers can still use their own timers if necessary, by setting the timeout argument in the send method. Below is an example response:

```
<response success="false" code="FP_NO_ANSWER_NETWORK" status="0">
</response>
```

This error can be simulated by disabling the Ethernet port on the fiscal printer for example (SET 19 CONNESSIONE set to NON CONNESSO).

6.10 Responses when Printer Offline

If the cover is open or no paper is in the printer, the FpMate CGI service returns an appropriate error. Below is an example response:

```
<response success="false" code="EPTR_REC_EMPTY" status="3">
        <addInfo>
                <lastCommand>88</lastCommand>
                <printerStatus>30110</printerStatus>
        </addInfo>
</response>
```

6.11 Printer Power Supply Interruption

Power supply interruption whilst there are no requests for the printer	Power supply interruption during FpMate CGI fiscal firmware communication exchanges
No effect. After power is restored to the printer the system is ready as before.	The FpMate CGI service executes on the fly using static memory and cannot recover from a power failure. Regardless of whether the service is used or not, if local printing was in progress, the fiscal printer upon restoration performs the following (except for management documents): • Cuts the paper • Prints a power interruption message • Prints again up to the same point This behaviour is not suitable when using this library. It is only suitable when employing the previous generation of Epson drivers or the POS keyboard. Most probably, no response will arrive at the browser and the XMLHTTP request will timeout. Since in most cases the FpMate CGI service automatically performs a printer reset when a request arrives, it is only necessary to send the same request again after the timeout has elapsed. The original print-out will be cancelled and will not be added to the daily totals

7 Fiscal-ePOS-Print API

The JavaScript library (filename fiscalprint.js) provides an interface for communicating with the FpMate CGI service. The library has the following key components:

Component	**Code**	**Description**
Constructors	epson.fiscalPrint	Initializes an ePOS-Print XML Builder object
Methods	send	Sends a message adding SOAP header. Four arguments can be passed with the last two being optional: • URL • Data • Timeout • callMode (async or sync) If timeout is not specified, 0 (no timeout) is assumed. If callMode is not specified async is assumed
Properties	onreceive	Response message event
	onerror	Communication error event

7.1 Practical Usage

- The first step is to insert the script line at the head of the HTML page:

```
<script  type="text/javascript" src="fiscalprint.js"></script>
```

- Let's create a queryPrinterStatus request. Firstly, we place the request in a variable:

```
var data_to_send = '<printerCommand>\n' +
          '\t<queryPrinterStatus  operator="1" />\n' +
          '</printerCommand>\n';
```

- Instantiate the ePOS request:

```
var epos = new epson.fiscalPrint();
```

- Place the onreceive code first:

```
epos.onreceive = function (res, tag_list_names, add_info, res_add) {
```

Place any code here to manage the responses.

```
} // end onReceive
```

- Place the onerror code after:

```
epos.onerror = function () {
```

Place any code here to manage errors. XMLHTTP request timeouts fire this event as well as HTTP errors. Printer errors on the other hand use onreceive.

```
} // end onError
```

- Now we can send the request:

```
epos.send(window.location.protocol + "//" + window.location.hostname + "/cgi-bin/fpmate.cgi",
data_to_send, timeout);
```

or with query string:

```
epos.send(window.location.protocol + "//" + window.location.hostname + "/cgi-bin/fpmate.cgi"
 + "?devid=cucina&timeout=10000", data_to_send, timeout);
```

8 UTF-8 Encoding and Character Escaping

The JavaScript and/or HTML files should be UTF-8 encoded since each request will have the following first line automatically inserted by the library:

```
<?xml version="1.0" encoding="utf-8" ?>
```

Italian Epson printers including fiscal variants use OEM codepage-437. The UTF-8 and code page standard ASCII characters are the same. However, non-standard characters such as grave accent "è" are represented by different codes. The FpMate CGI service automatically converts the following non-standard characters from UTF-8 to OEM CP-437 and vice-versa:

Ä ä à â
Ç ç
É è é ê ë
ì ï î
Ö ò ô ö
Ü ù ü û
ÿ
£ *
°
€ *

* The pound sign in certain cases automatically displays or prints the Euro symbol.

The Euro symbol can be both printed and displayed. In all cases the fiscal printer SET 15/16 value must be set to the default of 156.

The use of other non-standard characters could have undesirable effects.

In addition, the following characters must be represented with the appropriate HTML codes:

Character	HTML Escape Code
<	&lt;
>	&gt;
&	&amp;
"	&quot;
'	&apos;

9 Automatic Programming

Before processing certain types of XML file, the FpMate CGI service reads and sets some fiscal printer parameters.

9.1 printerFiscalReceipt, printerNonFiscal, printerFiscalReport and printerFiscalDocument File Types

The following sequence is used:

1. Sends reset command (closes any open document or other nor ready condition)
2. Reads SET 14/29 JAVAPOS-UPOS flag
3. If JAVAPOS-UPOS mode is deactivated (value zero), it activates it
4. Programs SET 14/11 flag to 1 (paper low warning on request)
5. Programs SET 14/27 flag to 0 (ACK off)
6. Processes XML file
7. if the original JAVAPOS-UPOS mode was deactivated, the setting is restored (Flag SET 14/29 set to zero)

9.2 printerCommand and printerCommands File Type

1. Programs SET 14/11 flag to 1 (paper low warning on request)
2. Programs SET 14/27 flag to 0 (ACK off)
3. Processes XML file

9.3 printerTicket

No preprograming is performed.

10 Native Command Cross Reference

XML Element	Native Command(s)
authorizeSales	1-084
beginFiscalDocument	1-052 and 1-089
beginFiscalReceipt	1-085
beginNonFiscal	1-063
beginTraining	4-014
clearText	1-062
displayText	1-062
EFTPOSDailyClosure	1-139
EFTPOSGetCurrentTotal	1-139
endFiscalDocument	1-087
endFiscalReceipt	1-087
endNonFiscal	1-065
endTraining	4-014
getDate	4-201
openDrawer	1-050
printBarCode	1-075
printContentByDate	3-103
printContentByNumbers	3-104
printDuplicateReceipt	1-047
printGraphicCoupon	n/a
printNormal	1-064
printRecCash	1-031, 1-032, 1-039 and 1-040
printRecItem	1-080
printRecItemAdjustment	1-083 and 1-090
printRecItemAdjustmentVoid	1-027
printRecItemVoid	1-027 and 1-082
printRecLotteryID	1-135
printRecMessage	1-078
printRecRefund	1-081 and 1-090
printRecRefundAdjustment	1-081 and 1-090
printRecRefundVoid	1-027
printRecSubtotal	1-086
printRecSubtotalAdjustment	1-083
printRecSubtotalAdjustVoid	1-027
printRecTotal	1-084
printRecVoid	1-028
printXReport	2-001

XML Element	Native Command(s)
printXZReport	3-002
printZReport	3-001
queryContentByDate	3-103
queryContentByNumbers	3-104
queryPrinterStatus	1-074 and 1-138
resetPrinter	1-088
setDate	4-001
setLogo	4-015

11 XML Examples

11.1 Commercial Document

```
<printerFiscalReceipt>
   <displayText operator="1" data="Message on         customer display" />
   <printRecMessage operator="1" messageType="1" index="1" font="1" message="First Additional Header Row Type 1" />
   <printRecMessage operator="1" messageType="1" index="2" font="1" message="Second Additional Header Row Type 1" />
   <beginFiscalReceipt operator="1" />
   <printRecMessage operator="1" messageType="4" message="First Additional Row Type 4" />
   <printRecItem operator="1" description="PANINO" quantity="1" unitPrice="6,00" department="1" justification="1" />
   <printRecMessage operator="1" messageType="4" message="Second Additional Row Type 4" />
   <printRecItem operator="1" description="Selling Item 2 VAT 22%" quantity="1,234" unitPrice="10,00" department="1" justification="1" />
   <printRecItemVoid operator="1" description="Void selling Item 2" quantity="1,234" unitPrice="10,00" department="1" justification="1" />
   <printRecItem operator="1" description="Selling Item 3 VAT 22%" quantity="2,50" unitPrice="100,17" department="1" justification="1" />
   <printRecMessage operator="1" messageType="4" message="Third Additional Row Type 4" />
   <printRecMessage operator="1" messageType="7" message="123456789" comment="Customer Id" />
   <printRecItem operator="1" description="Selling Item 4 VAT 10%" quantity="12,13" unitPrice="216,17" department="2" justification="2" />
   <printRecItemAdjustment operator="1" description="Discount applied to the product" adjustmentType="0" amount="123,45" justification="2" />
   <printRecMessage operator="1" message="First Additional Row Type 2" messageType="2" index="1" font="1" />
   <printRecMessage operator="1" message="Second Additional Row Type 2" messageType="2" index="2" font="1" />
   <printRecMessage operator="1" message="First Additional Row Type 3" messageType="3" index="1" font="1" />
   <printRecMessage operator="1" message="Second Additional Row Type 3" messageType="3" index="2" font="1" />
   <printRecItem operator="1" description="Selling Item 5 4%" quantity="12,13" unitPrice="216,17" department="3" justification="2" />
   <printRecSubtotalAdjustment operator="1" description="Discount applied to the subtotal" adjustmentType="1" amount="300,12" justification="2" />
   <printRecSubtotal operator="1" option="1" />
   <printBarCode operator="1" position="10" width="2" height="66" hRIPosition="3" hRIFont="C" codeType="CODE39" code="0123456789" />
   <printRecTotal operator="1" description="Payment in cash" payment="0" paymentType="0" index="1" justification="2" />
   <displayText operator="1" data="Message on         customer display" />
   <endFiscalReceipt operator="1" />
</printerFiscalReceipt>
```

11.2 Commercial Document with Lottery Code

```
<printerFiscalReceipt>
   <displayText operator="1" data="Message on         customer display" />
   <beginFiscalReceipt operator="1" />
   <printRecItem operator="1" description="PANINO" quantity="1" unitPrice="6,00" department="1" justification="1" />
   <printRecLotteryID operator="1" code="ABCDEFGN" />
   <printRecTotal operator="1" description="Payment in cash" payment="0" paymentType="0" index="1" justification="2" />
   <displayText operator="1" data="Message on         customer display" />
   <endFiscalReceipt operator="1" />
</printerFiscalReceipt>
```

11.3 Commercial Document with Deposit Deduction (Acconto)

```
<printerFiscalReceipt>
    <displayText operator="1" data="Message on        customer display" />
    <beginFiscalReceipt operator="1" />
    <printRecItem operator="1" description="TELEVISION" quantity="1" unitPrice="650" department="1"
    justification="1" />
    <printRecItemAdjustment operator="1" description="DEPOSIT ADJUSTMENT" adjustmentType="10" amount="100"
    department="1" justification="1" />
    <printRecTotal operator="1" description="Payment Bancomat" payment="550" paymentType="2" index="1"
    justification="1" />
    <displayText operator="1" data="Message on        customer display" />
    <endFiscalReceipt operator="1" />
</printerFiscalReceipt>
```

11.4 Commercial Document with Free of Charge Deduction (Omaggio)

```
<printerFiscalReceipt>
    <displayText operator="1" data="Message on        customer display" />
    <beginFiscalReceipt operator="1" />
    <printRecItem operator="1" description="ROUTER WI-FI" quantity="1" unitPrice="650" department="1"
    justification="1" />
    <printRecItemAdjustment operator="1" description="FREE OF CHARGE ADJUSTMENT" adjustmentType="11"
    amount="100" department="1" justification="1" />
    <printRecTotal operator="1" description="Payment Bancomat" payment="550" paymentType="2" index="1"
    justification="1" />
    <displayText operator="1" data="Message on        customer display" />
    <endFiscalReceipt operator="1" />
</printerFiscalReceipt>
```

11.5 Commercial Document with Single-use Voucher (Buono Monouso)

```
<printerFiscalReceipt>
    <displayText operator="1" data="Message on        customer display" />
    <beginFiscalReceipt operator="1" />
    <printRecItem operator="1" description="WATER FILTERS" quantity="12" unitPrice="4" department="1"
    justification="1" />
    <printRecItemAdjustment operator="1" description="SINGLE-USE VOUCHER" adjustmentType="12" amount="10"
    department="1"  justification="1" comment="Voucher for specific VAT or Nature" />
    <printRecTotal operator="1" description="Payment Bancomat" payment="38" paymentType="2" index="1"
    justification="1" />
    <displayText operator="1" data="Message on        customer display" />
    <endFiscalReceipt operator="1" />
</printerFiscalReceipt>
```

11.6 Z Report (Daily Closure)

```
<printerFiscalReport>
    <displayText operator="1" data="Message on        customer display" />
    <printZReport operator="1" timeout="30000" />
</printerFiscalReport>
```

11.7 Management Document with Bar Codes

```
<printerNonFiscal>
    <displayText operator="1" data="Message on        customer display" />
    <beginNonFiscal operator="1" />
    <printNormal operator="1" font="1" data="Management Document Row N. 1 Font 1" />
    <printNormal operator="1" font="2" data="Management Document Row N. 2 Font 2" />
    <printNormal operator="1" font="3" data="Management Document Row N. 3 Font 3" />
    <printBarCode operator="1" position="901" width="2" height="66" hRIPosition="1" hRIFont="A" codeType="CODE39"
    code="0123456789" />
    <printNormal operator="1" font="4" data="Management Document Row N. 4 Font 4" />
    <displayText operator="1" data=" Message on       customer display" />
    <printNormal operator="1" font="1" data="Management Document Row N. 5 Font 1" />
    <printNormal operator="1" font="2" data="Management Document Row N. 6 Font 2" />
    <printNormal operator="1" font="3" data="Management Document Row N. 7 Font 3" />
    <printBarCode operator="1" position="901" width="2" height="66" hRIPosition="ABOVE" hRIFont="B"
    codeType="CODE128" code="{A0123456789" />
    <printNormal operator="1" font="4" data="Management Document Row N. 8 Font 4" />
    <displayText operator="1" data=" Message on        customer display" />
    <endNonFiscal operator="1" />
</printerNonFiscal>
```

11.8 Direct IO Command

```
<printerCommand>
    <directIO command="4005" data="012200" comment="Set VAT group 1 to 22%" />
</printerCommand>
```

11.9 Commercial Document with Graphic Coupon

```
<printerFiscalReceipt>
    <printRecMessage operator="1" message="First Additional Header Row Type 1" messageType="1" index="1"
    font="1" />
    <beginFiscalReceipt operator="1" />
    <printRecItem operator="1" description="Selling Item 1 VAT 10%" quantity="1" unitPrice="100,00" department="2"
    justification="1" />
    <printRecMessage operator="1" message="Second Additional Row Type 4" messageType="4" index="1" />
    <printRecItem operator="1" description="Selling Item 2 VAT 22%" quantity="1,234" unitPrice="100,00"
    department="1" justification="1" />
    <printRecMessage operator="1" message="First Additional Row Type 2" messageType="2" index="1" font="1" />
    <printRecMessage operator="1" message="First Additional Row Type 3" messageType="3" index="1" font="1" />
    <printRecItem operator="1" description="Selling Item 5 22%" quantity="12,13" unitPrice="216,17" department="1"
    justification="2" />
    <printRecSubtotalAdjustment operator="1" description="Discount applied to the subtotal" adjustmentType="0"
    amount="300,12" department="3" justification="2" />
    <printRecSubtotal operator="1" option="1" />
    <printBarCode operator="1" position="901" width="2" height="66" hRIPosition="3" hRIFont="A" codeType="CODE39"
    code="01234567ABCDEF" />
    <printRecTotal operator="1" description="Payment in cash" payment="0" paymentType="0" index="1"
    justification="2" />
    <printGraphicCoupon operator="1" graphicFormat="R" comment="prints HELLO THERE"
    >SEVMTE8NCIRIRVJF</printGraphicCoupon>
    <displayText operator="1" data="Customer Display    Printed Fisc Receipt" />
    <endFiscalReceipt operator="1" />
</printerFiscalReceipt>
```

11.10 Management Document with Bar Codes

```
<printerNonFiscal>
    <beginNonFiscal  operator="1" />
    <printNormal  operator="1" font="1" data="Management Document Row N. 1 Font 1" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="1" data="Barcode System CODE39" />
    <printBarCode  operator="1" position="900" width="2" height="66" hRIPosition="2" hRIFont="A" codeType="CODE39"
    code="0123456789ABCD" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="2" data="Management Document Row N. 2 Font 2" />
    <printNormal  operator="1" font="3" data="Management Document Row N. 3 Font 3" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="1" data="Barcode System CODE93" />
    <printBarCode  operator="1" position="901" width="2" height="66" hRIPosition="1" hRIFont="A" codeType="CODE93"
    code="0123456789ABCD" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="4" data="Management Document Row N. 4 Font 4" />
    <printNormal  operator="1" font="1" data="Management Document Row N. 5 Font 1" />
    <printNormal  operator="1" font="2" data="Management Document Row N. 6 Font 2" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="1" data="Barcode System CODE128 CODE A" />
    <printBarCode  operator="1" position="902" width="2" height="66" hRIPosition="0" hRIFont="A"
    codeType="CODE128" code="{A0123456789ABCD" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="3" data="Management Document Row N. 7 Font 3" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="1" data="Barcode System CODE128 CODE B" />
    <printBarCode  operator="1" position="0" width="2" height="66" hRIPosition="1" hRIFont="A" codeType="CODE128"
    code="{B0123456789ABCD" />
    <printNormal  operator="1" font="1" data="" comment="Add blank line (whitespace)" />
    <printNormal  operator="1" font="4" data="Management Document Row N. 8 Font 4" />
    <displayText  operator="1" data="Customer Display Printed Non Fisc Doc" />
    <printNormal  operator="1" font="1" data="" />
    <printNormal  operator="1" font="1" data="Barcode System CODE128 CODE C" />
    <printBarCode  operator="1" position="123" width="2" height="66" hRIPosition="3" hRIFont="A"
    codeType="CODE128" code="{C01234567890123" />
    <displayText  operator="1" data="Customer Display    Printed Non Fisc Doc" />
    <endNonFiscal  operator="1" />
</printerNonFiscal>
```

11.11 Management Document with Graphic Coupon

```
<printerNonFiscal>
    <beginNonFiscal  operator="1" />
    <printNormal  operator="1" font="1" data="COUPON PRINTING ..." />
    <printGraphicCoupon  operator="1" graphicFormat="B" >askjdhasdjhkjasd................../=</printGraphicCoupon>
    <printNormal  operator="1" font="1" data=" COUPON PRINTED" />
    <endNonFiscal  operator="1" />
</printerNonFiscal>
```

11.12 Fiscal Document Invoice) Based on Last Commercial Document

```
<printerFiscalDocument>
    <printRecMessage  operator="1" message="First Add Header Row Type 5" messageType="5" index="1" />
    <printRecMessage  operator="1" message="First Add Header Row Type 5" messageType="5" index="2" />
    <printRecMessage  operator="1" message="First Client Row Type 6" messageType="6" index="1" />
    <printRecMessage  operator="1" message="Second Client Row Type 6" messageType="6" index="2" />
    <beginFiscalDocument  operator="1" documentType="lastReceiptInvoice" documentNumber="0" />
    <displayText  operator="1" data="Customer Display    Printed Last Rec Fis" />
</printerFiscalDocument>
```

11.13 Direct Fiscal Document (Invoice)

```
<printerFiscalDocument>
    <printRecMessage  operator="1" message="First Add Header Row Type 5" messageType="5" index="1" font="1" />
    <printRecMessage  operator="1" message="First Add Header Row Type 5" messageType="5" index="2" font="1" />
    <printRecMessage  operator="1" message="First Client Row Type 6" messageType="6" index="1" font="1" />
    <printRecMessage  operator="1" message="Second Client Row Type 6" messageType="6" index="2" font="1" />
    <beginFiscalDocument  operator="1" documentType="directInvoice" documentNumber="0" />
    <printRecItem  operator="1" description="Selling Item 1 22%" quantity="22,11" unitPrice="1,333" department="1"
    justification="2" />
    <printRecTotal  operator="1" description="Payment by cheque" payment="300,00" paymentType="1"
    justification="2" />
    <displayText  operator="1" data="Customer Display    Printed Dir Fisc" />
    <endFiscalDocument  operator="1" />
</printerFiscalDocument>
```

11.14 printContentByDate – Must be Logged in

```
<printerCommand>
    <printContentByDate  operator="1" dataType="0" fromDay="24" fromMonth="6" fromYear="2021" toDay="25"
    toMonth="6" toYear="2021" />
</printerCommand>
```

11.15 printContentByNumbers – Must be Logged in

```
<printerCommand>
    <printContentByNumbers operator="1" dataType="0" day="5" month="6" year="21" fromNumber="3" toNumber="7"
    />
</printerCommand>
```

11.16 queryContentByDate – Must be Logged in

```
<printerCommand>
    <queryContentByDate operator="1" dataType="0" fromDay="6" fromMonth="11" fromYear="2020" toDay="6"
    toMonth="11" toYear="2020" />
</printerCommand>
```

11.17 queryContentByNumbers – Must be Logged in

```
<printerCommand>
    <queryContentByNumbers operator="1" dataType="0" day="6" month="11" year="2020" fromNumber="23"
    toNumber="23" />
</printerCommand>
```

11.18 Printer Basic Status Query

```
<printerCommand>
    <queryPrinterStatus  operator="1" statusType="0" />
</printerCommand>
```

11.19 Printer RT Status Query

```
<printerCommand>
    <queryPrinterStatus  operator="1" statusType="1" />
</printerCommand>
```

11.20 Display Text Message

```
<printerCommand>
   <displayText operator="1" data="00 ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789A" />
</printerCommand>
```

11.21 Non-fiscal ePOS-print (for TM printers only)

```
<epos-print>
   <text dw="0" dh="0" description="Coupons" />
   <feed type="1" />
   <feed type="1" />
   <cut type="feed" />
</epos-print>
```

11.22 setLogo in Management Document

Logo uploaded, programmed in all four positions then set differently for commercial document footer

```
<printerNonFiscal>
   <beginNonFiscal operator="1" />
   <printNormal operator="1" font="1" data="LOGO UPLOAD AND PROGRAMMING ..." />
   <setLogo operator="1" location="0" index="1" option="0" graphicFormat="B" >123klkjaskdj…………/=</setLogo>
   <setLogo location="2" index="2" option="2" comment="Set commercial document footer to existing index 2" />
   <printNormal operator="1" font="1" data="LOG UPLOAD AND PROGRAMMING COMPLETE" />
   <endNonFiscal operator="1" />
</printerNonFiscal>
```

11.23 setLogo with printerCommand

```
<printerCommand>
   <setLogo location="3" index="3" comment="operator, option and graphicFormat attributes not necessary" />
</printerCommand>
```

11.24 RT Query Document Status

```
<printerCommand>
   <directIO command="9205" data="299MEY12345612012108880033000000000102" timeout="6000" />
</printerCommand>
```

11.25 RT Commercial Refund Document

If flag SET 14/58 = 0 / NO, replace printRecRefund with printRecItem.

Preferred method with REFUND keyword that includes fiscal serial number:

```
<printerFiscalReceipt>
    <printRecMessage  operator="1" message="REFUND 0279 0010 08012021 99MEY123456" messageType="4" />
    <beginFiscalReceipt  operator="1" />
    <printRecRefund operator="1" description="TV" quantity="1" unitPrice="600,00" department="1" justification="1" />
    <printRecTotal  operator="1" description="Payment refunded" payment="600,00" paymentType="0" index="1"
    justification="2" />
    <endFiscalReceipt  operator="1" />
</printerFiscalReceipt>
```

The older method with RESO MERCE N. keyword should no longer be used.

11.26 RT Commercial Void Document (Automatic Print-out)

Preferred method with VOID keyword that includes fiscal serial number:

```
<printerFiscalReceipt>
    <printRecMessage  operator="1" message="VOID 0279 0010 08012021 99MEY123456" messageType="4" />
</printerFiscalReceipt>
```

The older method with ANNULLAMENTO N. keyword should no longer be used.

11.27 RT Commercial Void Document (Manual Print-out)

If flag SET 14/59 = 1 / SI, replace printRecItem with printRecRefund.

Preferred method with VOID keyword that includes fiscal serial number:

```
<printerFiscalReceipt>
    <printRecMessage  operator="1" message="VOID 0279 0010 08012021 99MEY123456" messageType="4" />
    <beginFiscalReceipt  operator="1" />
    <printRecItem  operator="1" description="DVD" quantity="1" unitPrice="12" department="15" justification="1" />
    <printRecTotal  operator="1" description="Payment cancelled" payment="20" paymentType="0" index="1"
    justification="2" />
    <endFiscalReceipt  operator="1" />
</printerFiscalReceipt>
```

The older method with ANNULLAMENTO N. keyword should no longer be used.

11.28 Login

```
<printerCommand>
    <directIO  command="4038" data="0212345                                                                                               "
    comment="Login password 12345 followed by 95 spaces for a length of 100" />
</printerCommand>
```