planet

3cXml "ECR-POS" Protocol Reference Specification

Version 3.2.0

Document issue status

Revision date	Version	Author	Change
01 Mar 2004	1.0	Christian Schreiner	Initial draft
29 Mar 2004	1.1	Christian Schreiner	Added communication methods
19 May 2004	1.2	Jacques Hensen	EMV messages added
02 Nov 2004	1.3 (b-1)	Danny Heinen Alain Philipin	Complete review of document General comments and integrator considerations EMV scenario descriptions More explanations, examples
10 Nov 2004	1.3 (b-2)	Danny Heinen	Additional fields ZoneId, InvoiceId, AttendantId More integrator recommendations EftSettlementEmv - Sale and Refund
10 Nov 2004	1.3 (b-3)	Danny Heinen	Modified and more integrator recommendations
11 Nov 2004	1.3	Danny Heinen	Some more integrator comments and corrections after revision 3C.
12 Nov 2004	1.4 (b-1)	Danny Heinen	Card input method "E" missing in EftSettlementEmv.
08 Feb 2005	1.4	Jan Nielsen	Release of 1.4 – several changes related to version 1.3
04 May 2005	1.4.1	Jan Nielsen	Interim version with Cardholder AVS Fields added
01 Sep 2005	1.4.2	Christopher Danby	Removal of "EftDccAuthorizationSettlementEmv" message Update to Parking recommendations. Addition of "EftProcessDataIntegratorEmv" message.
30 Mar 2006	1.4.3	Christopher Danby	PCI overview Modification of messages to accomidate PCI
10 Feb 2009	1.4.3.1	Christopher Danby	Changes in DCC messages
26 Aug 2009	1.4.4.	Karnig Derderian, Danny Heinen	Added some missing fields and sizes in reply messages Extension of BankResultCode field to 4 digits Some more notes in chapter 4.1 AmountExtra field (was specified as not used)
05 Apr 2011 08 Apr 2011 30 Nov 2011 02 Dec 2011	1.4.5	Philipin Alain Danny Heinen Danny Heinen Danny Heinen	New EftTerminalControl commands: " closeOpenShift" "getEmvSettings" "getBinRanges" Remove DoPinBypass command. New messages types EftData New EftTerminalControl command "notify" Note about masked card numbers Remove some fields. New field EmvTerminalRefManufacturer Remove attendant list "AL" EMV scenario New field <EmvTerminalIdManufacturer> <Data1> additional information in EftTerminalStatus Remove old integrator recommendations Complete review of general considerations Add abbreviated fields
12 Mar 2012	1.4.6.	Danny Heinen Pascal Homann	Some fields missing in messages. Added message types EftTerminalEmv "Completion-Terminal" and EftProcessDataIntegratorEmv "PreAuth-Confirm". Some contactless related information added.

22 Mar 2012 23 Mar 2012	1.4.7	Danny Heinen	Message EftData is implemented now. Optional DCCFlag in EftProcessDataIntegratorEmv. <Token> field added in different messages.
25 Apr 2012	1.4.8	Danny Heinen	Some typo corrections
02 Oct 2013	1.5.0	Danny Heinen	DCCFlag in EftAuthorizationEmv message. EftAuthorizationEmv and EftSettlement support non EMV processing by getting card number or track. Add fields in EftAuthorizationEmv. Batch file format for batch processing New CardCheckEmv message New CardInputMethod for specific EMV case EmvTerminalId and EmvScenarioId optional
16 Oct 2013 22 Oct 2013 06 Nov 2013	1.5.2	Danny Heinen Danny Heinen Danny Heinen	In CardCheckEmv fields CardInputMethod, CardNumber and CardExpiryDate optional, depending on type. Real example flows, swipe, DCC and non DCC Some corrections with CVV, AVS fields not N/I.
21 Nov 2013	1.5.3	Danny Heinen	New field MarketData for market specific data added.
01 Feb 2014 12 Feb 2014 17 Feb 2014	1.5.4	Danny Heinen Danny Heinen Danny Heinen	New fields PinBlk and PinKsn for online pin processing. Remove field PrinReceipt. New fields EmvTerminalRefManufacturer and EmvTerminalKernelVersion. CVV2Result and AVSResult. New EftProcessDataIntegratorEmv messages types "Sale-Reversal", "Refund" and "Refund-Reversal" Field EmvTerminalRefManufacturer mandatory. Review EftProcessDataIntegratorEmv mandatory fields. Remove some obsolete text. Remove fields CardUsage and CardUsagePossibility, never used so far. In EftProcessDataIntegratorEmv review optional/mandatory fields. Add field TimeStamp in message EftTerminalstatus. Add field DCCMarkup in messages DCCCheck, EftAuthorizationEmv and EftSettlementEmv. New EftProcessDataIntegratorEmv types Preauth-Offline, Sale-Offline only used for offline transactions. New field < ConfirmRequired> in EftProcessDataIntegratorEmv Integra architecture diagram. Some more description in general chapters. New EftTerminalControl command "initialize" Review Market Data tags. Remove EftTerminalStatus "E" (error information) and "R" (terminal ready information), as never implemented so far. Release version.
17 Mar 2014	2.0.0-draft	Danny Heinen	Renamed protocol 3cXml to SIXml. New fields for 3DS processing <3ds...> New market data section for tags used in the lodging area New fields for Canada terminal using EftProcessDataIntegrator new fields <Mac> and <UpdateKeysRequired>

			New command "getKeys" in EftTerminalControl
14 May 2014	2.0.0	Danny Heinen	Typos in Market Data section. New fields for support of SRED. Add new field <KsnCardData >, rename field <PinKsn> into <KsnPin>. Add fields <EmvTrack2Equivalent> and <KsnCardData> in message CardCheckEmv for SRED support.
19 May 2014	2.0.0	Danny Heinen	New field DisableReceipt in EftTerminalControl "initialize" message. New parameter "disableCardCheckOnline" in EftTerminalControl "activate" <Data> field. Rename fields starting with 3ds... to Tds..., as not XML conform.
27 Jun 2014	2.0.1-draft	Danny Heinen	New sections for ECR and AUTH use cases, and specific integrated solution types. Split document up into multiple sub-documents to be able to generate different variants of the specification. TerminalId mapping to location.
17 Jul 2014		Danny Heinen	Change in the MarketData field. Remove first 2 bytes from tag name, and add Type attribute.
24 Jul 2014		Danny Heinen	Some changes in MarketData fields. New types added.
29 Jul 2014		Christophe Demaeght	3cXml specs for use case AUTH.
13 Aug 2014		Danny Heinen	New fields and messages for EP2 terminal implementations, to be integrated with MPD clients. New message EftCommit. New fields in EftAuthorizationEmv and EftSettlementEmv - ContractIndex - CommitRequired - CommitTimeout - CommitPartial - EmvTerminalVerifResult - EmvTransactionStatusInformation - TimeStamp (in reply) - CardSchemeLabel - CardholderLanguage - CardNumberEncrypted New field EmvApplicationId in message CardCheckEmv. New fields <PrintData1> and <PrintData2> in messages ShiftClose, ShiftOpen, ShiftCloseOpen, EftTerminalControl commands shiftClose and closeOpenShift. New fields EmvTerminalVerifResult and EmvTransactionStatusInformation
17 Nov 2014		Danny Heinen	
19 Nov 2014		Danny Heinen	Corrections in EftCommit, and take out specific EP2 terminal implementations comments.
21 Nov 2014		Danny Heinen	Some corrections related to EftCommit message fields. Accentuate RequesterTransRefNum auto generation.
24 Nov 2014	2.0.1	Danny Heinen	New fields ActSeqCounter and TransSeqNum for EP2 solution. New field EmvDedicatedFileName.

10 Dec 2014	2.0.2-draft	Danny Heinen	Field ActSeqCounter in EftTerminalControl commands "shiftOpen", "closeOpenShift" instead of "activate". New EftTerminalControl commands "reconciliation" and "reconciliationWithClosure". Optional field AttendantId in shiftClose message. Optional field ClosureSeqCounter in EftSettlementEmv reply.
8 June 2016	3.0.0	Danny Heinen	Re-branding to 3C. Remove EP2 protocol messages: EftCommit, EftTerminalcontrol commands "reconciliation" and "reconciliationWithClosure". Remove EP2 protocol fields: ActSeqCounter, ClosureSeqCounter, TransSeqNum, CommitRequired, CommitTimeout, CommitPartial. Add new EftAuthorizationType "FinalAuth". Add new field PosEnvironment in EftAuthorizationEmv and EftSettlementEmv. New authentication fields ValidationId and ValidationCode
1 July 2016	3.0.1	Danny Heinen	Some additions in lodging market data. CardInputMethod "P" for proximity (contactless) cards. Typical message types used in modes "ECR-POS" and "ECR-HOTEL". Added type FinalAuth-Reversal
13 June 2018	3.02	C. Demaeght	- Removed Transaction Token Support - Change description 4.8, RequesterLocationId is a must. - New Header attribute "RequesterHardwareId" - New fields "EmvFormFactorIndicator", "EmvMerchantCustomData" and "EmvTagsContainer". - New set of Final Auth related messages - Added "F" AuthCodeInputMethod.
24 May 2019	3.03	C. Demaeght	- Introduction of new 3CXml ECR attribute "TransRepeat" for Host-to-Host integration architectures (5.2.1). - Support of Satged Digital wallets (Alipay, Wechat, ...) in the 3CXml-ECR messages, with new fields PayMethod, ProductType and TransactionUID added in the EftSettlementEmv type (5.5.6). The description of the field Token is also updated (5.5.6).
01 August 2019	3.0.4-Draft	C. Demaeght	New fields "PayMethod" and "ProductType" replaced by existing fields respectively "CardInputMethod" and ""CardSchemeID". Data values remain the same as defined in v3.0.3.
02 August 2019	3.0.5	C.Demaeght	Adding field OriginalRTRN. Adding new 3CXml fields to support PSD2 mandatres: "TransInitiator", "COFIndicator", "MITType", "SCATransRef", "AgreementRef", "TdsVersion", "TdsDSTransID". Change "StatusCode" field side to 4 Digits, from 3-digits
22 October 2020	3.1	C. Demaeght	Added new 3CXml field "SCAExemptionInd". Added 2 new Message Types GetToken and GetPAN (Bulk)
09 Juillet 2021	3.2	C. Demaeght	Added new optional field "PosVersion" in 3CXml Request messages Added new structure of fields "TaxFreeData" in 3CXml Request and Response messages.

Table of contents

Terms

Acquirer	Institution which might be a bank, but not necessarily, which is accepting authorization and settlement requests for cards.
Authorization (or final authorization)	The process of holding an amount for a merchant, until the merchant clears the transaction (settlement), or the hold "falls off" after some time. First a validation of the card number is done, then the Integra application might do an online authorization (to the bank) or approve the request locally, in case the amount is lower than a floor limit. The amount of the request shall be the final (settlement) amount.
Authorization code	A code that an issuer or its authorizing processor provides to indicate approval or denial for an authorization request.
BIN (of a card)	The bank identification number of a card. The first six digits of a Visa or MasterCard account number. This number is used to identify the card-issuing institution.
Channel	Communication method which can be a tcp/ip direct communication, http/https or SOAP interface.
Credit (transaction)	A transaction where an amount is credited to the customer's bankcard account.
DCC	Dynamic Currency Conversion. For DCC eligible cards a currency conversion rate is provided.
Debit (transaction)	A charge to a customer's bankcard account.
EFT	Electronic funds transfer.
EMV card	Integrated chip card integrating an application corresponding to the EMV standard
EMV standard	Europay – Mastercard – Visa standard related to credit card applications on chip cards.
Floor limit	The amount above which the Integra application is doing an online authorization. In case the amount in the request is below this value a local authorization done.
ICC	Integrated chip card
Integra	Name of the Planet server application. Integra can be a locally installed server, hosted in our datacenter or be an EMV terminal.
Integrator	The merchant system implementing the interface to the Integra application. This can be a restaurant POS system, a parking system, a hotel PMS or a retail POS.
Invoice company	The bank finally settling the transaction created by the Planet system.
Issuer	Any association member financial institution, bank, credit union or company that issues, or causes to be issued, plastic cards to cardholders.
Merchant	An entity that contracts with merchant banks to originate transactions.
PAN (of a card)	Card primary account number. The number that is embossed and/or encoded on a plastic card that identifies the issuer and the particular cardholder account.
PED	Pin entry device
PIN	Personal identification number. A sequence of digits used to verify the identity of the holder of a token. The PIN is a kind of password.
PMS	Property management system.
POS	Point of sales
Pre-Authorization	The process of holding an amount for a merchant, until the merchant clears the transaction (settlement), or the hold "falls off" after some time. First a validation of the card number is done, then the Integra application might do an online authorization (to the bank) or approve the request locally, in case the amount is lower than a floor limit. The amount of the request must not be the final (settlement) amount. It might be increased by using top-up (incremental) authorizations.
Referral	Response received in an online authorization request, indicating that request has been rejected, but also the merchant should contact the acquirer to eventually get a manual authorization code.
Refund	See Credit transaction.
Requester	Integrator.
Settlement	The action of finalizing a card transaction on the Integra application. This transaction will be sent to the Planet processing system and from there to the invoice company.
Topup - Authorization	The process of incrementing the amount hold for a merchant, until the merchant clears the transaction (settlement), or the hold "falls off" after some time.
Validation	Operation of checking a card against the validity of the card number and expiry, start date. No amount taken into account.

1 Introduction

1.1 Purpose

This document provides a reference of all the protocol messages used between an integrator solution such as restaurant, hotel, retail, parking or other system, and the Integra solutions.

The Integra application may need to connect to other parties like an EMV terminal, the Authorization server, directly to the banks, to our service hosts, or to or DCC hosts or others to be able to process the payment processing requests.

> **The document here does provide some rough guidelines for different integrated solutions**, but the main intention is to provide a reference catalogue to have a complete overview of all messages existing.
>
> **In all cases a close contact to integration engineering team is required to discuss details of the solution, how to integrate, interconnect and what payment processing flows shall be used. Specific guidelines for all type of solutions can be provided at request.**

1.2 How to use this document

The 3cXml protocol may appear very complex if one looks at all the messages existing and the different possibilities with some of them. There are administration messages, status, credit card or EMV messages and some other messages. The reason for this is the diversity of the systems using this protocol. Some of the systems have a rather simple functionality, others are more complex, and need more services from Integra and the our hosts.

> An integrator must not implement all the messages provided by 3cXml. The final implementation depends on the actual integrator solution and the needs of the merchant.

1.3 3cXml specification variants

As different integrated solution types exist and can be provided by us, different variants of the 3cXml specifications exist. The integration engineer will provide you the one which is relevant for you.

> **!?!** **The 3cXml variant of the document here contains descriptions for the POS integrated solution types when using the 3cXml protocol in an "ECR-POS" use case.**

2 The Integra solution

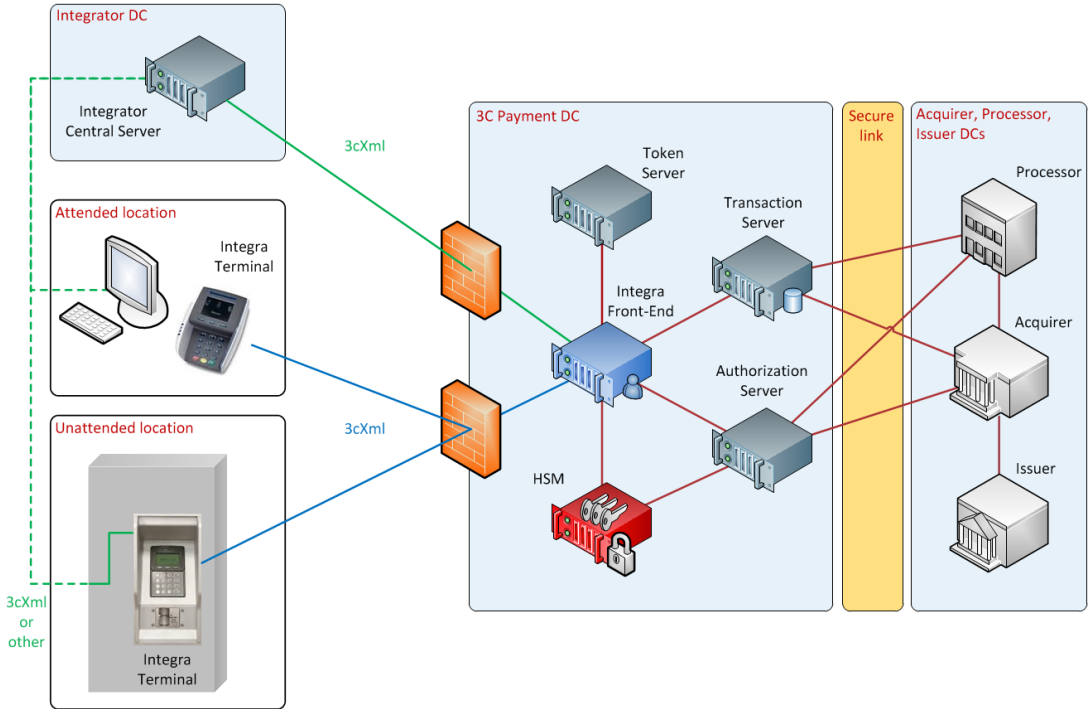
2.1 Introduction

The Integra solution provides card payment services towards external parties, also called integrators. These integrators are typically solutions which require the possibility to process Electronic Founds Transaction (EFT) payments via credit and debit cards, such as property management systems (PMS) for hotels, point of sales (POS) solutions for the restaurants and retails market, parking solutions or vending machines. Integra provides multiple interface possibilities for such integrators to easily process card payments.

Operations possible via these interfaces are:

- Card validation
- Card reading
- Card EFT authorisation and settlement
- DCC requests
- Working shift operations
- EMV terminal management
- EMV configuration data reception

The following diagram provides an overview of the architecture of the Integra solution.



The 3cXml interface protocol supports usage of certified EMV terminals to process the payment, and still supports classical swipe card handling where the card is read at the integrator site, and sent to Integra Front-End (note, this is not recommended anymore).

The 3cXml protocol might be used between the integrator and the Integra Front-End (Integra-FE) server or directly between the integrator and the EMV terminal. The Integra-FE server might communicate to an Integra terminal (also using 3cXml) or other vendor EMV terminal.

All external communication links are secured by using SSL channels. Note, it is also possible to install the Integra-FE at the customer location, e.g. in a hotel, restaurant or parking facility. This was the original deployment mode, but not really promoted anymore, due to PCI reasons.

2.2 Integrated Solutions types and use cases

The 3cXml protocol is flexible and can support different Integrated Solution types:

- Type POS: an **attended solution used in restaurant, retail or parking business**, where the integrator system communicates directly to an EMV terminal or via a (Front-End) central server. Most of times rather simple payment messages are needed to support SALE, REFUND, REVERSAL operations.

- Type HOTEL: an **attended solution used in hospitality or car rental business**, where the integrator system communicates to an EMV terminal via a (Front-End) central server. More payment messages are needed to support more complex flows and including PRE-AUTH, FINAL-AUTH, TOP-UP, COMPLETION, SALE, REFUND, REVERSAL operations. Also DCC messages are used in this type of solution.

- Type VENDING: an **unattended solution used in vending and parking business**, where the integrator system communicates directly to an EMV terminal or via a (Front-End) central server. More payment messages are needed to support more complex flows and including PRE-AUTH, FINAL-AUTH , TOP-UP, COMPLETION, SALE, REFUND, REVERSAL operations. In addition TERMINAL CONTROL and STATUS are needed to support more complex flows for the interaction between the unattended station and the EMV terminal.

- Type PETROL: an **unattended solution used in petrol business**, where the integrator system communicates directly to an EMV terminal or via a (Front-End) central server. More payment messages are needed to support more complex flows and including PRE-AUTH, FINAL-AUTH, TOP-UP, COMPLETION, SALE, REFUND, REVERSAL operations. In addition TERMINAL CONTROL and STATUS are needed to support more complex flows for the interaction between the unattended station and the EMV terminal. Specific messages are needed to support PETROL functionalities.

In these solutions the 3cXml protocol can be implemented in the EMV terminal itself, in an intermediate component or on a hosted (Front-End) Integra central server. Two main use cases of the protocol exist:

- Use case "ECR": 3cXml is used by the integrator to communicate to an EMV terminal, in order to process payment operations and get the status of the terminal. 3cXml is used as **terminal communication interface protocol**.

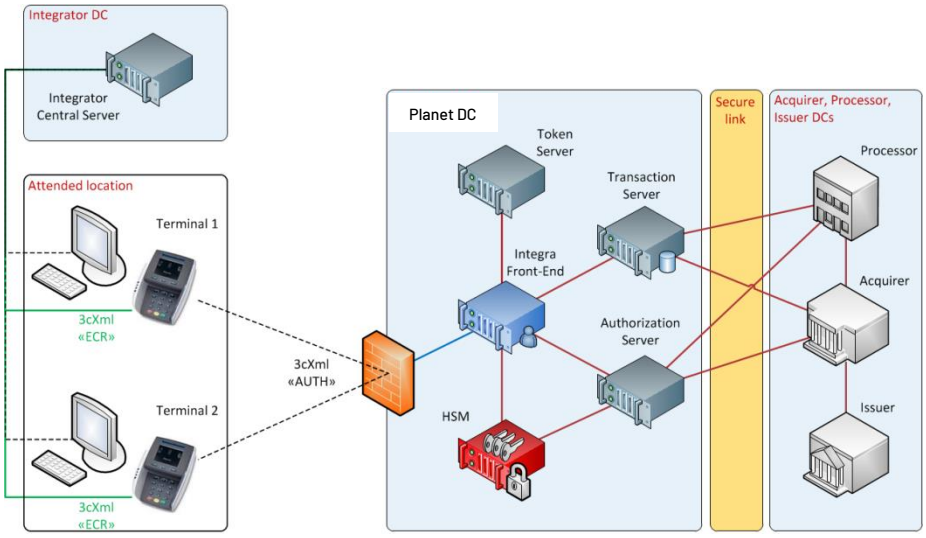
- Use case "AUTH": the integrator controls the EMV terminal or card reader, or collects the card data by another way, and uses 3cXml as **authorization interface protocol**. All the card data is provided in the 3cXml message for further authorization towards an acquirer.

Different solution types and use cases are more detailed in the next chapter, depending on the 3cXml specification variant you currently have.

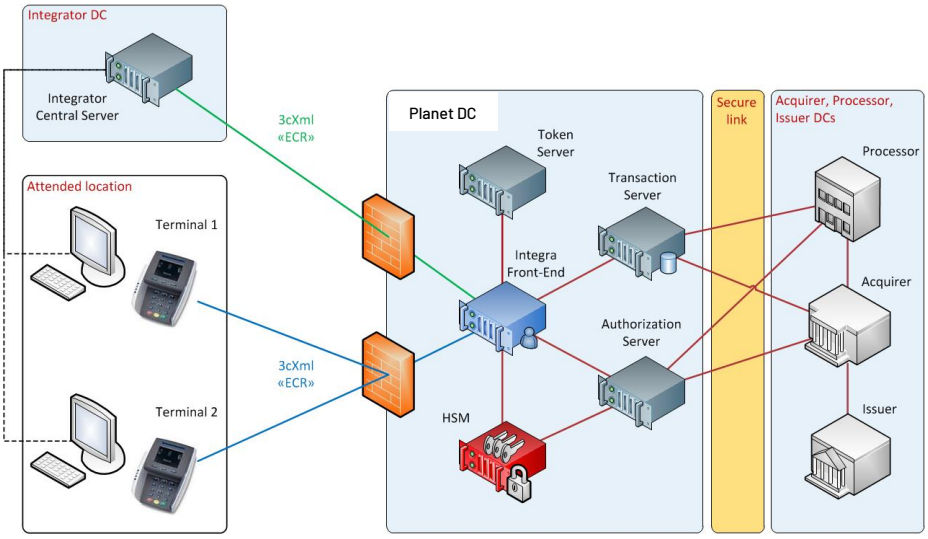
2.3 Integrated solution mode “ECR-POS”

An integrated solution of mode ECR-POS might be an attended retail solution, an attended parking solution (the cardholder goes to e.g. in case of lost ticket), or a restaurant solution. Rather simple payment transactions have to be supported, like a direct SALE or REFUND transaction.

The integrator solution has either a direct connection to the EMV terminal or via a central server. The following diagrams show possible deployments.



The green line shows the usage of 3cXml as "ECR" protocol. High level payment transaction requests are processed via that connection. The EMV terminal does an authorization to a central Integra Front-End host, using 3cXml too, but in the AUTH use case.



In this diagram the integrator is "connected" to the terminals via a central Front-End server. The integrator POS server connects to the Integra Front-End server and this one connects to the terminals to be used, either by using 3cXml, or any other protocol. The payment messages going to the central server must include a terminal identification to determine what terminal shall be used.

Overview of typical messages used in this mode "ECR-POS"

For payment processing:

- EftSettlementEmv – type "Sale-Terminal": to process a direct sale (debit) purchase transaction
- EftSettlementEmv – type "Refund-Terminal": to process a direct refund (credit) transaction
- EftSettlementEmv – type "Sale-Reversal": to void a previously created sale transaction
- EftSettlementEmv – type "Refund-Reversal": to void a previously created refund transaction

- Cancel: to abort a payment request

For administration (optional):

- ShiftClose: to close a reconciliation shift
- ShiftOpen: to open a reconciliation shift
- ShiftCloseOpen: to close, then re-open a reconciliation shift

For terminal management (optional):

- EftTerminalControl – command "initialize": to initialise an EMV terminal (ip settings, language)
- EftTerminalStatus: to retirve the status of an EMV terminal

Other messages (optional):

- CardCheckEmv: read out card data (masked) and provide a card token
- EftData: receive the data of a formerly processed transaction

3 Message types overview

\\3cint.local\Files\Luxembourg\DEV\Home\lu3cchde\Documents\3C\Documents\3C Payment\Architecture & Engineering\Specifications\3CXml\3CXml-ECR\3CXml-ECR v3.1.0 (ECR) - PLANET\3cXml-specs-Message types overview ECR-POS.docx

3.1 General messages types

The following list shows the general message types of the 3cXml protocol, non-payment processing related.

Message	Type	Description

CheckStatus	Status	Status request. Returns if the application is operational or not.
ShiftOpen	Admin	Open the reconciliation period.
ShiftClose	Admin	Close the reconciliation period.
ShiftCloseOpen	Admin	Close then open the reconciliation period.
Cancel	Cancel	Cancel a message sent before.
Error (reply)	Error	An error in the request, before determining the request type.

3.2 Terminal messages types

The following list shows the terminal related message types of the 3cXml protocol, used for specific EMV terminal handling.

Message	Type	Description
EftTerminalControl	Terminal	Control an EMV terminal connected to Integra.
EftTerminalStatus	Terminal	Provides status information about the EMV terminal.

4 3cXml protocol general considerations

4.1 DCC handling

4.1.1 With EMV terminal

In most solutions where an EMV terminal is involved dynamic currency conversion display and selection is supported by the usage of the terminal. If enabled the DCC currency and rate is displayed in the terminal screen and the cardholder is requester to enter a choice. The choice is sent back in the corresponding response message.

4.1.2 With no EMV terminal (obsolete)

When no EMV terminal is used and the card data is read and provided by the integrator the message DCCCheck should be used to get the BIN currency and the daily rate of the card, if the card is DCC eligible. The DCC currency and rate are displayed at the integrator's workstation screen, and the DCC choice is made there. A DCCCheck shall be made prior an authorization and Settlement request to determine the rate of the date. In the authorization or settlement request the integrator specifies if DCC is chosen or not, by setting the DCCFlag field.

4.2 References

4.2.1 Payment cycle

References are important and must be used when multiple requests are used with the target to process a final EFT payment. Reference must not be used when a payment can be done in a single request, such as single Sale or Refund request.

A payment cycle is a sequence of payment requests such as initial authorization (pre-auth), supplemental ones (top-up) and payment completion, and possibly a further payment request (sale) when the cardholder has left the location. Payment cycles are used e.g. in hotels where top-up requests are used during the stay of the guest, or in parking systems where the card is read at the entrance and the payment amount is determined with the same card at exit. But they can also be used e.g. in the car rental industry where a pre-authorization is done at car checkout and a completion when the card is returned.

Whenever a payment cycle is used in the target solution it is important that a reference is included in the final payment transaction request.

Two kinds of references exist in 3cXml and can be used depending of the solution. Refer to our project engineering team to choose the right reference mechanism for the solution. They are described in the next sections.

> When using the reference the card number must not be sent back and forth in every exchange, which is important in the PCI aspect. Like this the card number can be completely excluded in the messages.

4.2.2 RequesterTransRefNum

The most common payment cycle reference currently used is the **RequesterTransRefNum** header attribute. This is the integrator reference of the payment and it is passed in every request related to a payment.

It is for instance used to find back initial and supplemental requests concerning one transaction. For some acquirer we have to cancel the initial authorization in case we make a supplemental one. Depending on the integrator application, this can be a "hotel reservation id", or "restaurant check number", or a parking "parking ticket number". Whenever a customer pays with the same credit card for two payment transactions the reference should be different.

A new payment cycle starts with the first pre-authorization done with using a new RequesterTransRefNum.

Notes:

- The RequesterTransRefNum does also appear in the reports we are providing to the integrator, so double interest in using it.

> It is possible to configure Integra Front-End or the terminal to **auto generate a unique RequesterTransRefNum** for every new start of payment cycle (pre-auth request), when not existing in the request. The RequesterTransRefNum can then be used by the integrator for all subsequent requests.

4.2.3 Token

The token is a reference built in Integra and can be used to reference upcoming requests linked to an initial one. The token can be transaction related or card related. The card related tokens are card PAN substitutes, but not encrypted card numbers, so cannot be decrypted. From a card token itself it is not possible to generate back the initial card number used to build the token. So they are secure in the PCI data security perspective.

Different kinds of tokens exist, and must be configured in Integra.

4.2.3.1 Transaction token

Not supported anymore.

4.2.3.2 Card token

A card token is substitute (replacement) number for an existing card number (PAN). Depending on the algorithm used, a card token can be a fully ISO compatible credit card number passing a Luhn validation check. A card number with an official Planet bin range can be generated.

> When a **card token** is used the same token value will be sent back for all payment requests. So it is not really a unique reference for a payment cycle, and it is on the integrator to build its internal payment reference. As an example a parking with a card in card out solution would not send a RequesterTransRefNum at drive in, but receive a token, and at drive out receive the same token to calculate the amount to be used for the payment.

Note that card tokens are not generated locally in Integra, but in our central token host. Card tokens can be used in multiple locations. So it would be possible to generate a card token in a hotel for a reservation, and use it during its validity in other hotels of the same chain.

Notes:

- In obsolete non EMV requests the token functionality can also be used, but with the difference that the initial card reading is not done on an EMV terminal, but on the integrator's swipe card reader. A token can be generated at initial pre-auth and from that point on used as reference in subsequent requests. Doing this the card number is only sent once over the network.
- Different token algorithms exist. Verify with our project engineering team what the best algorithm is for the solution.

The RequesterTransRefNum and card token can be used in parallel. So the RequesterTransRefNum is used within a payment cycle, but the card token is still returned, and always the same for the same card. The card token can then be used to process late charged, as with a real card number.

4.3 EMV card authorization and settlements with EMV terminals with Integra

4.3.1 Introduction

The best way to process credit card transactions is to use EMV certified terminals for card reading (chip, swipe or contactless), possible manual card number entry and PIN entry.

The main mechanisms to handle EMV cards with Integra are the following ones.

4.3.2 All EMV terminals are connected to Integra FE

In this modus Integra receives EMV authorization and settlement requests without containing any card data. Integra communicates with the EMV terminal and the terminal provides the required EMV data. The authorization and transaction processing is then done using the terminal and the integrator gets back some limited card (e.g. masked PAN, token), EMV information and transaction receipt.

The field **<EmvTerminalId>** is filled with the terminal ID configured in Integra. Integra will use this EMV terminal to read out the card data. This scenario will is named "*DT*" (Direct Terminal). For this scenario the messages **EftAuthorizationEmv or EftSettlementEmv** have to be used.

We can also have the situation that only one EMV terminal is connected to the Integra server, which might be installed e.g. as an application on a till. In this case no terminal ID must be sent in the request. This scenario is named "*DU*" (Direct Unique). The field **<EmvTerminalId>** can be left empty.

The field <ScenarioId> must be filled in all EMV messages to indicate the scenario to be used.

> In case the **"DT"** scenario is used, the **<EmvTerminalId>** passed to Integra-FE can be freely chosen by the integrator. But it is important that the terminal Ids are correctly setup in the Integra-FE server application, in coordination with the integrator. Otherwise the terminal cannot be found.
>
> Instead of filling the field **<EmvTerminalId>** the field **<RequesterStationId>** can be used in combination with **<RequesterLocationId>.** Integra will then internally map the <RequesterStationId> and <RequesterLocationId> to a defined terminal. The field **<EmvTerminalId>** can must still exist, but can be left empty in this case.

4.3.3 The integrator handles the EMV terminal

In this modus the integrator handles the EMV terminal and includes all EMV data required in the EMV request. Towards an Integra Front-End server. The Integra FE is used as an authorization host. The authorization or transaction creation is then done by Integra FE and the result is sent back to the requester. The Integrator has an EMV terminal attached to his own system and he will provide us all EMV data needed to Integra to process the EMV online authorization or transaction storage. This scenario will be named "*DI*" (Data Integrator), and only the message **EftProcessDataIntegratorEmv** is required.

4.3.4 The integrator communicates to EMV terminal

In this modus the integrator communicated directly to an Integra terminal, and not to an Integra FE. The terminal in this case communicates towards an Integra FE. The authorization or transaction creation is then done by Integra FE and the result is sent back to the terminal, which sends back to the requester.

4.3.5 Capabilities of EMV terminals

Integra has integrated EMV terminals of many brands and flavours. But the end user functionality (e.g. **wireless, printing, gratuity, referral, transaction selection** ...) is depending on the EMV terminal used. Some terminals might implement a function differently than other ones, or not preview it at all. Contact project engineering team to get more information on the EMV terminals and their functionality included.

4.4 Receipt printing

The cardholder and merchant receipts can be printed out on the EMV terminal, **depending on the terminal type used**. Not all terminals provide a printer.

When the terminal cannot be used for the receipt printing they can also be printed out on the integrator solution. In the response of authorization and settlement messages Integra formats and returns the receipts in the fields **<PrintData1>** and **<PrintData2>**. The <PrintData1> contains the merchant receipt and the field <PrintData2> the cardholder receipt. These receipts should be printed out on the integrator solution as received from Integra. Depending on the terminal solution they come directly from the certified EMV terminal, and must not be changed. In case one of the fields is empty the integrator should make nothing is printed out, such as an empty page.

4.5 Status messages

3cXml has the possibility to provide status messages of the ongoing transaction on a specific terminal. The following information can be sent back:

- Terminal card status (inserted, removed, card data)
- Terminal display message (text shown on the terminal screen, possibly translated into country language)
- Terminal power status
- Terminal information, general status

The status messages are sent back at real time while waiting for the final response. For most solutions the use of status messages is optional. But it is always good practice to handle status messages, e.g. to provide feedback for the attendant of the status of the operation.

In certain solutions such in the unattended payment solutions it is mandatory to use the status messages.

4.6 User data fields

The user data fields <UserData1> and <UserData2> in the settlement requests can be used by the integrator solution to store its own information (e.g. internal references). These fields will appear in our transaction reports provided to the customer.

4.7 Connection mode, queuing and multiple requests

> The current chapter is applicable only for the communication towards the Integra FE.

It is advised to open a new connection for every new request to be done, and close the connection when the response has been sent back.

It is possible to leave the connection permanently open. In this case status messages should be sent to regularly check the availability of the server.

Integra can support message queuing on the same TCP/IP socket channel. This means that the requester can send in a new EFT request before the answer of the preceding one is received. However, when this is done the RequesterTransRefNum field of the request must be different in the new request.

The requester can also open multiple TCP/IP socket channels at the same time. And thus do a new request on the newly opened socket. Like this he can also do multiple requests the same time.

> Integra is by default not setup for message queuing and this function needs to be enabled. Before using the function in live environment is important to carefully test it, as it is more complex then opening a new connection for every new transaction to be processed.

The choice of the connection mode depends on the integrator solution and should be discussed with Planet. With a centrally hosted Integra-FE server it is recommended the integrator uses HTTPS.

4.8 Multiple locations per channel

For Integra a location is a customer property, such as a parking location, a restaurant or a hotel. Integra is generally configured in that way to accept requests for multiple locations on the same channel. Therefore the field RequesterLocationId must be used to indicate the location to be used for the transaction. This RequesterLocationId can be chosen by integrator, in this case it has to be communicated to Planet at setup/installation time, so that it can be mapped by configuration to the internal Integra location Id.

In addition to the RequesterLocationId a RequesterStationId can be sent in the request. Integra is able to map the combination of RequesterLocationId and WorkstationId to its internal location Id. This feature can for instance be used for parking systems which to map a payment station to a configured location in Integra.

In addition Integra can find a location by using the TerminalId field from the request. In this case all terminalsIds used by the integrator MUST be configured in the Integra location, and every terminal is dedicated mapped to one location only. This is not the recommended setup and it MUST be agreed with Planet.

4.9 Communication channel and data link

Even when using TCP/IP it is advised to use a communication data link with ACK/NACK handshake to make sure that messages correctly arrive at Integra. Even with TCP/IP as communication channel we have seen situations where messages had been sent by a requester, but never arrived at Integra, due to routing or firewall settings. Checksum calculation is not required when using TCP/IP, as it takes care of the integrity of the message.

4.10 Timeout and cancel

The Integra application implements several timeouts to recognize error conditions:

- Not able to get an outgoing channel for a specific time
- Not able to connect to authorization server
- No response received from authorization server within certain time

As soon as one of these timeouts occur the Integra application will return an error. These timeouts are set to be adapted to the channels used for the authorization.

Nevertheless each integrator should implement its own security timeout, after which it cancels the request. It is important that the integrator timeout is high enough, to not interfere with the Planet timeout. Even if the authorization channel is very fast it can be slowed down in some situations. As an extreme example you could take an application which uses in a normal situation a fast multiple ADSL Internet channel (3-5s authorization time), but then in an error condition is switched down to a single (queued) PSTN modem channel (30-40 s). In this situation the authorizations would still go through, but would be much slower. Recommended timeout is 90 seconds for POS and PMS systems while Parking and Retail integrators can utilize down to 30 seconds in the case Integra is setup as a pure TCP/IP environment.
A timeout is a balance between correctness and efficiency and we will accommodate any special timeout as long as it can be clearly defined and well-reasoned.

4.11 Reconciliation period

The Integra application internally works with a reconciliation, or shift period. This is later used to group transactions together. When the close of the reconciliation period is requested (Shift Close), all transactions not in a closed state are going to be closed with the requested date.

Rules:

- Transactions have to be **closed** for them being **uploaded** to the Planet transaction host. This means if the shift is not closed they are not uploaded and stay on the Integra system, and thus **are not charged**.
- The shift has to be **open** for the Integra application to **accept a settlement** request.
- The shift open and close commands can be sent at any time.

In general the integrator once a day sends a close shift message then an open shift right after, or uses the combined close/open shift message. The transactions within a reconciliation period then are grouped together in the reports provided by us.

> It is not absolutely necessary to implement the close and open shift messages. The Integra application can be setup to automatically close and open the shift at a certain time. Whether to implement the function is a question of the utilization of the integration system. A restaurant POS might reconcile from 16:00 to 03:00, while parking systems reconcile by end of day. The parking systems generally do not need to integrate the shift messages, and the restaurant POS systems normally do.

4.12 3cXml abbreviated protocol

For those systems where the data link is not fully Ethernet network based and network performance is low (e.g. GPRS or serial) there is an "abbreviated" version of the 3cXml protocol existing. The protocol is the same, but all the xml node and attribute names have been shorten to reduce the data to be transferred as much as possible. Refer to appendix **"3cXml abbreviated protocol"** for more information.

4.13 Other considerations

- Always make sure the shift is open when a settlement is done.
- In case an authorization has been done, and is finally not used (customer pays by cash, or by another card) the authorization should be cancelled. To do this the corresponding **"Reversal"** message should be sent to the Planet server.
- It is important that the **<Result>** field in the settlement reply is taken into account. In case this one is different to "A" the settlement has not been done. The **<ResultReason>** is secondary information from which you can deduct the reason of especially rejected transactions.
- It is good practice to display the **<Message>** as a minimum when transactions are rejected, and dependent on the <ResultReason> several options can appear. (Other payment method, manual Authorization via phone or possibly retry if it's a system error).

5 Detailed message description

5.1 Syntax

5.1.1 XML request and reply fields

- The XML fields in the request and reply can be placed in any order.
- All fields should contain only ASCII chars.

5.1.2 Convention in the field descriptions

- In the request description the fields surrounded by [ ] are optional. If not otherwise specified all other fields are mandatory.
- Date definitions: y=year, M=month, d=day, H=hour, m=minute, s=second. A full date would then be "yyyyMMddHHmmss".
- A "/" in a field definition indicates a choice of different options.
- Message definitions are formatted in red italic characters

5.1.3 Abbreviations in field descriptions

Field types:

- N = numeric
- AN = alpha numeric
- B = boolean 1 or 0
- Y/N = Y (yes) or N (no)

Field presence in message:

- O = Optional (field may not be present or can be empty)
- M = Mandatory (field must be present and contain a value)
- O/M = Optional or mandatory. Depends on the situation clarified in the description.
- N/I = Not implemented. Currently not implemented by Integra application. Check with Planet regarding the implementation status.
- N/U = Not used. Currently not used. Check with Planet regarding its implementation.
- N/A = Not applicable.

5.2 Commonly used fields in most messages

5.2.1 Request and reply header

All 3cXml messages must contain some header attributes in the request and reply root node. Some of them are mandatory, some of them are not.

Field Name	M/O (*)	Max length (bytes)	Description
Type	M	/	Determines the **main type** of request. Subtypes of the request are sent in the message body.
SequenceNumber	M	10	A unique number for the request. The same number should be sent back in the reply. This number has to be used to cancel this message with the cancel message.
RequesterTransRefNum	M	20	This is a unique number related to the Eft transaction payment cycle. This number should be the same for all requests related to a same transaction. Like initial authorization (PreAuth), supplemental authorizations (TopUp) and finally the settlement. The same number is send back in the reply.
RequesterLocationId	O	10	A reference to the requester's location making this request.
RequesterStationId	O	10	A reference to the requester's station (PMS station, pay on foot parking terminal) making the request.
ValidationId	O	64	A credential identifier, to authenticate the requester. Provided by the integrator or 3C.
ValidationCode	O	64	A credential password to authenticate the requester. Provided by 3C.
RequesterHardwareId	O	20	Unique identification of the hardware (EMV Terminal, Pinpad) used for the transaction. Commonly known as the MAC Address.
TransRepeat	O	10	Optional attribute which can be set with values "True" or "False" to indicate if the 3CXml Request is a repetition (TransRepeat="True") message.

(*) mandatory or optional

TransRepeat: this attribute is introduced to handle some communication specific issues in Host-to-Host types of ECR integration architectures. A good example of such issues is when our Integra application returns successfully a 3CXml response to the ECR (Integrator), but this response was never successfully received by the ECR.

In this situation, the sending of a "Cancel" message by the ECR after the original request has timed out, would generally be unsuccessful. Also the use of a reversal message is not recommended. Instead we recommend the ECR to send a repetition message.

A repetition message will be identified by the new 3CXml Header attribute "TransRepeat" all other data elements must remain strickty identical to the original transaction request.

"TransRepeat" values to consider:

- Absence of TransRepeat, TransRepeat="", TransRepeat="False": this is an original transaction request.
- TransRepeat="True": Repetion Message;
- TransRepeat with other value, the attribute will be ignored.

The recommandation to Integrators should be to trigger a repetition message only once, and only after timeout of the orignal transaction. The timeout to apply on the repetition message should also be lower than the timeout to the original request.

How will the repetition message be processed:

- If the original transaction was already completed (successfully or not), return the same answer as to the original request, but with TransRepeat="True".

- If the original transaction is still being processed, the response to the repetition message will be returned following completion;

- If no original transaction can be found, the repetition message will be processed as an original one.

5.2.2 Result field in reply

The **<result>** field in the reply is one byte long and informs about the successful processing of the request. The result is present in all replies. There are four result types existing.

A	Accepted	The request has been successfully processed and accepted. For instance a "A" in an authorization request approves the transaction. Any other value means the authorization has been rejected.
R	Rejected	The request has been rejected. For instance for a settlement request this byte means the settlement has not been created.
W	Warning	A Warning can be seen as rejected, but in some cases the integrator might react differently. For instance an authorization referral could be sent as Warning. The integrator might prompt the user to enter a manual authorization code.
E	Error	As rejected. An error is present in one the request fields or happened in the Integra application.

For some requests with this field the **<ResultReason>** field is sent in the reply. This one gives more information about the reason of failure.

> The **Result** field is the most important field in the reply. This is the ONLY field one which has to be taken into account for the determination of the successful procession of a request (internal operation or payment transaction). All other fields are additional information.

5.2.3 Result reason field

In all **authorization requests** not only the result of the operation is sent back in the reply, but also the reason of the result. This can be a locally taken decision, or a remote decision from an external authorization server.

It can be either a numeric or an alphabetic code. If alphabetic the reply comes from Integra and was not online processed. It could however be rejected by the terminal.

Alphabetic General codes

- XA: Rejected as an transaction has already been uploaded and cannot be cancelled
- XB: Rejected as bad card type
- XD: Rejected as duplicate transaction request
- XE: Rejected as internal error in the application. This might be due to a wrong configuration
- XF: Rejected as wrong card function requested
- XG: Rejected in case we don't find a DCC request when we search it in DB
- XH: Rejected as hot card
- XI: Rejected as invalid card number
- XJ: Rejected DCC request
- XK: Rejected if the card number in top-up is different to card number in pre-auth
- XO: Accepted, locally approved (floor limit)
- XR: Rejected for other reason
- XS: Rejected as settlement failed shift is closed (Open shift and retry)
- XU: Rejected as tried to call out for authorization but host unavailable (connection problem, protocol problem)
- XX: Rejected as card expired
- XZ: Might be locally rejected or accepted. If rejected then amount was above floor limit, and not allowed to call or no auth path existing. If accepted for a completion request it was above floor limit but an authorization exists.
- PF: Generated when transaction request pending not found
- RD: duplicate request
- SC: Generated in case DCC selection is requested

Alphabetic Terminal related codes

TB: Rejected Terminal Busy

- TC: Rejected as cancelled on terminal
- TE: Rejected as error on terminal
- TF: Rejected Terminal Not Found
- TO: Locally accepted by terminal (EMV terminal)
- TR: Rejected by terminal
- TX: Rejected Terminal not configured

Numeric codes

In case the code is numeric it might come directly from the bank or be rejected by an online request made from an EMV terminal. We can have two or four digits, depending of the bank. In general Apacs based authorization hosts return 2 digits, our 3CISO auth host return four digits.

In general a number equal to zero means accepted, and not equal to zero means rejected.

Finally it is important to note that these codes don't have to be handled. They just provide additional information. The field which always count is the <result> field. This one determines the acceptance of the request.

5.2.4 Amount fields in the messages

All amounts in the requests are positive. In case of negative operations the authorization message type changes: refund, reversal.
The dot decimal separator has to be used for floating point values. Examples: "12.45", ".05", "14", "15.9576".

5.2.5 Track 2 card data format

The following track 2 card data formats are accepted in the request fields:

- N…DyyMMSSSR…
- N…= yyMMSSSR…
- BN…DyyMMSSSR…FL
- ;N…= yyMMSSSR…?L

where

- N… is the card number
- D or = is the date delimiter
- yyMM is the expiry date
- SSS is the service code
- R… is the rest of track 2
- B or ; is the start sentinel
- F or ? is the end sentinel
- L is the LRC of track 2

5.2.6 BankResultCode

The bank result field is the result code we receive back from the acquirer or the EMV terminal in case of online authorisation. It should NOT be used to decide on the successful processing of a request. The **Result** field is used for that purpose (see above). In the past bank result codes not being zero were always a reject, but this is not true anymore. We today might receive bank result codes which are non-zero for approved authorizations, depending on the country and acquirer.

5.3 Message "CardCheckEmv"

5.3.1 Purpose

This request does a validation of a card or card data. This means the card number, expiry date and other card details are validated. It returns the card type, card Invoice company and the card token if configured. Also a masked card number can be returned. The CardCheckEmv message can be used to:

- Validate a card. If rejected Integra cannot process this card.
- Read the card.
- Get a card token for a specific card number.

Two use cases exist.

In case the card is read by the integrator the track2 or just the card number and expiry date is provided to in the message request. No EMV terminal is used. The card data is validated and a token is created, which is returned in the message response among other fields.

In case an EMV is used and controlled by Integra the card is read in the EMV terminal and then validated. Also a token is generated. In the response message the token and other fields are sent back. The tone can be used for later processing.

Important: when an EMV terminal is used with a ICC (chip) card after card insertion the card stays in the chip reader, even after the message is sent back to the integrator. The terminal does not propose to take out the card. This is required to be able to start a payment operation right after, and the cardholder has not to insert it again.

The CardCheckEmv message is in fact used to determine the card type and to decide what payment operation can be executed. For example in Germany Girocards do not support pre-authorization. Si in case a Girocard card type is detected and returned to the integrator, a Sale request is made, rather than a pre-authorization request.

Note, Integra can be configured to send back card data for certain ranges, e.g. for loyalty cards. Even if the card might be rejected by Integra it might then be processed by the integrator.

> Attention, the "CardCheckEmv" message cannot be provided for all EMV terminal types. Certain EMV terminals we work with do not support reading of the card. Check with Planet engineering to determine of the solution provided supports it.

5.3.2 Use cases

The message CardCheckEmv can be used with and without EMV terminal. The field *<Type>* determines whether a terminal shall be used or not.

5.3.2.1 EMV terminal scenario (DT, DU)

In case an EMV terminal is used process an EMV transaction only the **"DT"** (Direct Terminal) or "**DU**" (Terminal Unique) scenarios can be used. The authorization request is processed by a specific EMV terminal. The

integrator passes a terminal reference in the request. An error message is sent back in case the **<EmvScenarioId>** field is empty and set to another value than "DT" or "DU".

In case the **"DT"** scenario is used, the **<EmvTerminalId>** passed to the Integra server can be freely chosen by the integrator. But it is important that the terminal Ids are correctly setup in the Integra server application, in coordination with the integrator. Otherwise the terminal cannot be found.

Instead of using the field **<EmvTerminalId>** the field **<RequesterStationId>** can be used in combination with **<RequesterLocationId>** to determine the terminal to work with. **This needs to be configured in Integra.**

5.3.2.2 No EMV terminal scenario

For the processing of NON-EMV card checks, where the card is read at the integrator side the card data is passed in the request. Also a token only can be sent. The fields <EmvScenarioId> and <EmvTerminalId> are optional.

5.3.3 Request and reply

<Request
Type="CardCheckEmv"
SequenceNumber="10 bytes"
RequesterTransRefNum="20 bytes"
[RequesterStationId="10 bytes"]
[RequesterLocationId="10 bytes"]
[ValidationId="64 bytes"]
[ValidationCode="64 bytes"]>
<Type>40 bytes</Type>
[<CardInputMethod>12</CardInputMethod>]
[<CardNumber>20 bytes</CardNumber>]
[<CardExpiryDate>MMyy</CardExpiryDate>]
[<CardStartDate>MMyy</CardStartDate>]
[<CardIssueNumber>12</CardIssueNumber>]
[<CardTrack1Data>0 bytes</CardTrack1Data>]
[<CardTrack2Data>50 bytes</CardTrack2Data>]
[<CardTrack3Data>40 bytes</CardTrack3Data>]
[<EmvTerminalId>25 bytes</EmvTerminalId>]
[<EmvScenarioId>12</EmvScenarioId>]
[<EmvTrack2Equivalent>40
bytes</EmvTrack2Equivalent>]
[<KsnCardData >1234</KsnCardData>]
</Request>

<Response
Type="CardCheckEmv"
SequenceNumber="10 bytes"
RequesterTransRefNum="20 bytes"
[RequesterStationId="10 bytes"]
[RequesterLocationId="10 bytes"]
[ValidationId="64 bytes"]
[ValidationCode="64 bytes"]>
<Type>40 bytes</Type>
<Result>1</Result>
<ResultReason>123</ResultReason>
<CardNumber>20 bytes</CardNumber>
[<CardInputMethod>12</CardInputMethod>]
[<Token>40 bytes</Token>]
<CardExpiryDate>MMyy</CardExpiryDate>
<CardSchemeId>12</CardSchemeId>
<CardSchemeName>30 bytes</CardSchemeName>
<CardFunctionId>12</CardFunctionId>
<CardFunctionName>30 bytes</CardFunctionName>
<CardInvoiceCompanyId>12</CardInvoiceCompanyId>
<CardInvoiceCompanyName>30 bytes
</CardInvoiceCompanyName>
[<EmvApplicationId>32 bytes</EmvApplicationId>]
[<CardIssueNumber>12</CardIssueNumber>]
[<CardStartDate>MMyy</CardStartDate>]
[<Message>90 bytes</Message>]
</Response>

5.3.4 Type field values

The **<Type>** field determines if an EMV terminal shall be used or not to process the card check. The following values can exist.

5.3.4.1 "Card-Request"

Check the card data received in the request, and return any card data available and a token (if configured). No EMV terminal is used.

5.3.4.2 "Card-Terminal"

Use an EMV terminal to read the card, and validate the card. Return any card data available and a token (if configured). For a chip card the card stays inserted in the terminal and is ready for a next transaction request.

5.3.4.3 "Card-Terminal-Remove"

Use an EMV terminal to read the card, and validate the card. Return any card data available and a token (if configured). After the card has been read, the terminal requests to remove the card.

5.3.5 Fields

Pres	Type	Max len	Content
O/M	N	4	Card expiration date. Format MMyy.
N/U	A	2	Planet's card function definition like CC for credit card, CD for debit card... Currently not used.
N/I	AN	30	Card function name like "credit card", "emv card"
O/M	A	2	"E" for EMV chip (contact ICC), "P" for proximity (contactless NFC), "S" for swipe (track), "M" for manual entered.
M	A	2	Planet reference of Invoice company (bank): CI, CE, ...
M	AN	30	Name of invoice company; Citicorp, Cekab, ...
O/M	N	2	Card issue number. Optional in case the card has been swiped.
O/M	N	20	Card PAN. No start and end sentinel.
M	A	2	Contains the card scheme Id: VS, AX, MC, XX, ... Might contain special Planet defined card schemes for certain cards.
M	AN	30	Name of the card scheme like "Visa", "Mastercard", "Diners", ...
O	N	4	Card start date. Format MMyy.
O	N	40	Card track 1 raw data.
O/M	N	50	Card track 2 raw data. Mandatory in case the card is swiped.
O	N	40	Card track 3 raw data.
O	A	2	2 bytes Id to indicate the EMV scenario to choose. Default is DT.
O	AN	25	Reference of EMV terminal used to get EMV card details
M	AN	20	Reference of application use during an EMV transaction
O/M	AN	40	EMV track 2
O	AN	32	Card data encryption key serial number used to generate the encrypted card data. In case SRED is enabled on the terminal.
O	AN	90	Text message which can come from a remote authorization host or from the Integra application.
M	A	1	Result of the operation. A (accepted), R (rejected), E (Warning), ...
M	AN	2	Indicates the reason of the result. XB (local bad card type), ...
O	N	40	Token, instead of or in addition of card number.
M	N	40	Determines from where the card data comes. Either provided in request or read by an EMV terminal.

In case the terminal is SRED enabled, and provides encrypted card data the fields <CardNumber>, <CardTrack2Data> or <EmvTrack2EquivalentData> contain the encrypted data, and the field <KsnCardData> contains the DUKPT encryption key serial number.

5.3.6 Example

```
<Request
  Type="CardCheckEmv"
  RequesterLocationId="362005"
  SequenceNumber="1234"
  RequesterStationId="269304"
  RequesterTransRefNum="123456">
<Type>Card-Request</Type>
<CardInputMethod>M</CardInputMethod>
<CardNumber>41111.....1111</CardNumber>
<CardExpiryDate>0306</CardExpiryDate>
</Request>
```

```
<Response
  Type="CardCheckEmv"
  RequesterLocationId="362005"
  RequesterStationId="269304"
  RequesterTransRefNum="123456"
  SequenceNumber="1234" Type="CardCheckEmv">
<Type> Card-Request</Type>
<CardExpiryDate>0306</CardExpiryDate>
<CardFunctionId>CC</CardFunctionId>
<CardFunctionName>credit
card</CardFunctionName>
<CardInvoiceCompanyId>NT</CardInvoiceCompanyId
>
<CardInvoiceCompanyName>Natwest
</CardInvoiceCompanyName>
<CardNumber>41111.....1111</CardNumber>
<CardSchemeId>VS</CardSchemeId>
<CardSchemeName>VISA</CardSchemeName>
<Message>Approved floor limit</Message>
<Result>A</Result>
<ResultReason>XO</ResultReason>
<Token>6049742569087734</Token>
</Response>
```

5.4 Message "EftSettlementEmv"

5.4.1 Purpose

The message EftSettlementEmv is used to create an EFT payment transaction. It can for instance be used to directly create a new sale transaction (including an online authorization) or to do a completion of a former pre-authorization.

As for the EftAuthorizationEmv message an EMV terminal controlled by Integra can be used to read out the card, or the card data can be passed in the request fields. Also a token only can be passed, and Integra is searching back the card data from a former request.

5.4.2 Use cases

The message EftSettlementEmv can be used with and without EMV terminal. The field *<EftSettlementType>* determines whether a terminal shall be used or not. Whenever the string "-Terminal" is present in *<EftSettlementType>* a terminal is used.

5.4.2.1 EMV terminal scenario (DT, DU)

In case an EMV terminal is used process an EMV transaction only the **"DT"** (Direct Terminal) or **"DU"** (Terminal Unique) scenarios can be used. The settlement request is processed by a specific EMV terminal. The integrator passes a terminal reference in the request. An error message is sent back in case the **<EmvScenarioId>** field is empty and set to another value than "DT" or "DU".

> In case the **"DT"** scenario is used, the **<EmvTerminalId>** passed to the Integra server can be freely chosen by the integrator. But it is important that the terminal Ids are correctly setup in the Integra server application, in coordination with the integrator. Otherwise the terminal cannot be found.
>
> Instead of using the field **<EmvTerminalId>** the field **<RequesterStationId>** can be used in combination with **<RequesterLocationId>** to determine the terminal to work with. **This needs to be configured in Integra.**

5.4.2.2 No EMV terminal scenario

For the processing of NON-EMV transactions, where the card is read at the integrator side the card data is passed in the request. Also a token only can be sent. In this case the preauthorization is done as manual card entry authorization. The fields <EmvScenarioId> and <EmvTerminalId> are optional.

5.4.3 Request and reply

```
<Request
   Type="EftSettlementEmv"
   SequenceNumber="10 bytes"
   RequesterTransRefNum="20 bytes"
   [RequesterStationId="10 bytes"]
   [RequesterLocationId="10 bytes"]
   [ValidationId="64 bytes"]
   [ValidationCode="64 bytes"]>
<EftSettlementType>40 bytes</EftSettlementType>
[<EmvTerminalId>25 bytes</EmvTerminalId>]
[<EmvScenarioId>12</EmvScenarioId>]
[<EmvTerminalMessage>50 bytes</EmvTerminalMessage>]
[<PosVersion>40 bytes</PosVersion>]
[<InvoiceId>50 bytes</InvoiceId>]
[<ZoneId>20 bytes</ZoneId>]
[<AttendantId>20 bytes</AttendantId>]
 [<CardNumber>20 bytes</CardNumber>]
[<Token>40 bytes</Token>]
[<CardExpiryDate>MMyy</CardExpiryDate>]
[<CardInputMethod>12</CardInputMethod>]
<Amount>10 bytes</Amount>
[<Currency>]10 bytes</Currency>]
[<TransactionUID Type"">128 bytes</TransactionUID>]
[<AuthCodeInputMethod>12</AuthCodeInputMethod>]
[<BankAuthCode>123456</BankAuthCode>]
[<BankResultCode>1234</BankResultCode>]
[<ResultReason>123</ResultReason>]
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
[<AmountAuthorized>10 bytes</AmountAuthorized>]
[<AmountVAT>10 bytes</AmountVAT>]
[<AmountExtra>10 bytes</AmountExtra>]
[<CardTrack1Data>40 bytes</CardTrack1Data>]
[<CardTrack2Data>50 bytes </CardTrack2Data>]
[<CardTrack3Data>40 bytes</CardTrack3Data>]
[<CardStartDate>MMyy</CardStartDate>]
[<CardIssueNumber>12</CardIssueNumber>]
[<UserData1>20 bytes</UserData1>]
[<UserData2>20 bytes</UserData2>]
[<UserData3>20 bytes</UserData3>]
[<UserData4>20 bytes</UserData4>]
[<DCCFlag>Y|N</DCCFlag>]
[<LocalAmount>10 bytes</LocalAmount>]
[<LocalCurrency>123</LocalCurrency>]
[<BinRate>14 bytes</BinRate>]
[<BinAmount>10 bytes</BinAmount>]
[<BinCurrency>123</BinCurrency>]
[<CardholderNameFirst>255</CardholderNameFirst>]
[<CardholderNameLast>255</CardholderNameLast>]
[<CardholderStreetAddress1>255</CardholderStreetAddress1
>]
[<CardholderStreetAddress2>255</CardholderStreetAddress
2>]
[<CardholderCity>50</CardholderCity>]
[<CardholderZipCode>10</CardholderZipCode>]
[<CardholderState>50</CardholderState>]
[<CardholderCountry>50</CardholderCountry>]
```

```
<Response
   Type="EftSettlementEmv"
   SequenceNumber="10 bytes"
   RequesterTransRefNum="20 bytes"
   [RequesterStationId="10 bytes"]
   [RequesterLocationId="10 bytes"]
   [ValidationId="64 bytes"]
   [ValidationCode="64 bytes"]>
<Result>1</Result>
<ResultReason>123</ResultReason>
<BankResultCode>1234</BankResultCode>
<BankAuthCode>123456</BankAuthCode>
<BankTerminalId>30?</BankTerminalId>
[<CardNumber>20 bytes</CardNumber>]
[<Token>40 bytes</Token>]
[<CardExpiryDate>MMyy</CardExpiryDate>]
[<CardTrack1Data>40 bytes </CardTrack1Data>]
[<CardTrack2Data>50 bytes </CardTrack2Data>]
[<CardTrack3Data>40 bytes </CardTrack3Data>]
<CardInputMethod>12 </CardInputMethod>
<CardSchemeId>12</CardSchemeId>
<CardSchemeName>30 bytes</CardSchemeName>
<CardSchemeLabel>30 bytes</CardSchemeLabel>
<CardFunctionId>12</CardFunctionId>
<CardFunctionName>30 bytes</CardFunctionName>
<CardInvoiceCompanyId>12</CardInvoiceCompanyId>
<CardInvoiceCompanyName>30 bytes
</CardInvoiceCompanyName>
<Amount>10 bytes</Amount>
[<TransactionUID Type"">128 bytes</TransactionUID>]
[<AmountExtra>10 bytes</AmountExtra>]
<AmountUsed>10 bytes</AmountUsed>
[<BinAmount> 10 bytes </BinAmount>]
[<BinCurrency> 123 </BinCurrency>]
[<BinRate> 12 bytes </BinRate>]
<DCCFlag>Y|N</DCCFlag>
<CurrencyUsed>10 bytes</CurrencyUsed>
<EftSettlementType>40 bytes </EftSettlementType>
<LocalAmount>10 bytes</LocalAmount>
<LocalCurrency>123</LocalCurrency>
<TransRefNum>20 bytes</TransRefNum>
<EmvPinVerified>Y/N</EmvPinVerified>
<EmvAuthorized>Y/N</EmvAuthorized>
[<EmvTerminalId>25 bytes</EmvTerminalId>]
[<EmvTerminalIdManufacturer>25
bytes</EmvTerminalIdManufacturer>]
[<EmvTrack2Equivalent>40 bytes
</EmvTrack2Equivalent>]
[<EmvApplicationId>32 bytes</EmvApplicationId>]
[<EmvTerminalVerifResult>10 bytes
</EmvTerminalVerifResult>]
[<EmvTransactionStatusInformation>4 bytes
</EmvTransactionStatusInformation>]
[<EmvPinVerified>Y/N</EmvPinVerified>]
[<EmvCryptogram>16 bytes</EmvCryptogram>]
[<EmvCryptogramType>12</EmvCryptogramType>]
```

[<CardholderCompany>50</CardholderCompany>]
[<CardholderEmail>50</CardholderEmail>]
[<CVV2>1234</CVV2>]
[<MarketData Type='2 bytes'>
<Data tag="6 bytes">255 bytes</Data>
<Data tag="6 bytes">255 bytes</Data>
...
</MarketData>]
[<TdsAuthentResultCode>20 bytes
</TdsAuthentResultCode>]
[<TdsAuthentIndicator>20 bytes </TdsAuthentIndicator>]
[<TdsAuthentCryptogram>255 bytes
</TdsAuthentCryptogram>]
[<TdsTransIdentifier>255 bytes </TdsTransIdentifier>]
[<PosEnvironment>1 </PosEnvironment >]
[<TransInitiator> 1 byte </TransInitiator>]
[<OriginalRTRN>20 bytes</OriginalRTRN>]
[<COFIndicator> 1 byte </COFIndicator>]
[<MITType> 1 byte </MITType>]
[<SCATransRef> 40 bytes </SCATransRef>]
[<AgreementRef> 40 bytes </AgreementRef>]
[<SCAExemptionInd> 1 byte </SCAExemptionInd>]
[<TdsVersion> 1 byte </TdsVersion>]
[<TdsDSTransID> 40 bytes </TdsDSTransID>]
[<TaxFreeData> XML Structure </TaxFreeData>]
</Request>

[<EmvUnpredictableNumber>8 bytes
</EmvUnpredictableNumber>]
[<EmvData1>40 bytes</EmvData1>]
[<EmvData2>40 bytes</EmvData2>]
[<CardHolderName>30 bytes</CardHolderName>]
[<CardHolderLanguage>30 bytes</ CardHolderLanguage
>]
[<MerchantId>20 bytes</MerchantId>]
[<CardIssueNumber>12</CardIssueNumber>]
[<CardStartDate>MMyy</CardStartDate>]
[<Token>40 bytes</Token>]
[<Message>90 bytes</Message>]
[<PrintData1>1000 bytes</PrintData1>]
[<PrintData2>1000 bytes</PrintData2>]
[<CVV2Result>1</CVV2Result>]
[<AVSResult>1</AVSResult>]
[<DCCMarkup>10 bytes</DCCMarkup>]
[<TimeStamp>yyyyMMddHHmmss</TimeStamp>]
[<PosEnvironment>1 </PosEnvironment >]
[<SCATransRef> 40 bytes </ SCATransRef>]
[<SCAExemptionInd> 1 byte </SCAExemptionInd>]
[<TaxFreeData> XML Structure </TaxFreeData>]
</Response>

5.4.4 EftSettlementEmv types (EmvSettlementType field)

5.4.4.1 "Sale-Terminal"

The same rules than for the normal settlement request applies. An EMV terminal is used to enter the card and eventually the PIN. Two scenarios can be used with the message, "DT" Direct Terminal or "DU" Terminal Unique. The selection of the appropriate scenario is done with the tag <**EmvScenarioId**>.

5.4.4.2 "Refund-Terminal"

Refund done by using an EMV terminal to read out the EMV data and optionally to enter a PIN. Two scenarios can be used with the message, "DT" Direct Terminal or "DU" Terminal Unique. The selection of the appropriate scenario is done with the tag <**EmvScenarioId**>. This is a direct **credit transaction**. The amount can have any value. The transaction can go for an online request in case the amount is higher than the Integra floor limit of the card.

5.4.4.3 "Sale"

This is a **debit transaction** request. In one request, the transaction is validated, authorized and settled. In case no authorization code is passed in the request the application can go for an online authorization, depending on the Planet's floor limit. When an authorization code is passed in the request the settlement is done offline. This request is trying to do an EMV settlement without terminal. No PIN entry is required. The Integra application will search and use the EMV parameters from the database, by using the requester transaction id. These ones might have been collected on a previous **"EftAuthorizationEmv"** (PreAuth) or **"EftSettlementEmv"** (Sale-Terminal). In case no EMV data is found the card information from the request is used. In this case a NON-EMV transaction which will be created.

5.4.4.4 "Refund"

This is a direct **credit transaction** request. The amount can have any value. The transaction can go for an online request in case the amount is higher than the Integra floor limit of the card.
Like the Sale request the Refund is getting the EMV information out of the Integra database. In case no EMV data is found the card information from the request is used. . In this case a NON-EMV transaction which will be created.

5.4.4.5 "Completion"

This request is for a **finalization of a transaction** for which pre-authorizations and top-up authorizations have been done before. The completion is done without EMV terminal. The same requester transaction reference number or token has to be passed in the request than for the initial and supplementary authorizations. The authorization code received from the bank must be supplied by the integrator. The Completion might go online depending on the bank interface.

5.4.4.6 "Completion-Terminal"

This request type is used for a **finalization of a transaction, but where the terminal is used,** e.g. to provide a screen with the DCC choice and confirmation, and/or to printout the receipt on the terminal, if configured so.

5.4.4.7 "Sale-Reversal"

Reversal of a sale. No PIN entry required. The requester transaction reference number has to be the same than in the Sale request.

5.4.4.8 "Refund-Reversal"

Reversal of a refund. No PIN entry required. The requester transaction reference number has to be the same than in the Refund request.

5.4.4.9 "Completion-Reversal"

Reversal of a completion. No PIN entry required. The requester transaction reference number has to be the same than in the Completion request.

5.4.5 Amount and currency of transaction

> For all requests **only the <Amount> and <Currency>** fields determine the amount and currency to start within that request. All other amount and currency fields are for information.
> However, during the processing the values might be changed, for instance if a tip is entered in a restaurant, or if the cardholder chooses another currency. The final real amount and currency processed are returned in the fields **<AmountUsed> and <CurrencyUsed>**.

5.4.6 Fields

Name	Pres	Type	Max len	Content
EftSettlementType	M	AN	40	Determines the settlement request type.
AgreementRef	O	ANS	40	Merchant Reference of the agreement between the Merchant and the Customer for storing the card credentials.
Amount	M	N	10	Amount of the request. See section **Error! Reference source not found.**.
AmountAuthorized	O	N	10	Amount which has been finally authorized.

AmountUsed	M	N	10	The actual amount used in the process. Amount, or total amount. In case a gratuity has been given this one is included.
AmoutExtra	O	N	10	An extra amount (e.g. gratuity). In case a gratuity is added by using an EMV terminal the amount entered is sent back in the reply.
AmountTotal	O	N	10	Total amount in local currency.
AmoutVAT	O	N	10	Amount VAT.
AuthCodeInputMethod	O/M	A	2	"A" for automatic, "M" for manual entry
AttendantId	O	AN	20	For future use. An identification of the operator or waiter. To associate requests to specific attendant.
AVSResult	O	AN	1	Result code of the address verification.
BankAuthCode	O	AN	9	Authorization code received from the acquirer.
BankResultCode	O	N	4	Result code from the bank in case of online authorization.
CardExpiryDate	O/M	N	4	Card expiration date. Format MMyy.
CardFunctionId	N/U	A	2	Planet's card function definition like CC for credit card, CD for debit card... Currently not used.
CardFunctionName	N/I	AN	30	Card function name like "credit card", "emv card"
CardholderCity	O	AN	50	For future use
CardholderCompany	O	AN	50	For future use
CardholderCountry	O	AN	50	For future use
CardholderEmail	O	AN	50	For future use
CardholderLanguage	O	AN	6	Language used by the cardholder. Format: "language-Country", e.g. "en-GBP".
CardholderNameFirst	O	AN	255	For future use
CardholderNameLast	O	AN	255	For future use
CardholderState	O	AN	50	For future use
CardholderStreetAddress1	O	AN	255	Used with AVS (Address Verification)
CardholderStreetAddress2	O	AN	255	For future use
CardholderZipCode	O	AN	10	Used with AVS (Address Verification)
CardInputMethod	M	A	2	Gives indication on how the card or payment mean was entered at the Point of Interaction (POI). In 3CXml response messages this indicates how the payment mean was entered at the POI: values are "E" for EMV chip (contact ICC), "P" for proximity (contactless NFC), "S" for swipe (track), "M" for manual entered, "W" for Staged Digital Wallets (SDW). In 3CXml requests messages, this is used to: - Indicate how how the card or payment mean was entered, when this is entered at the requester side (POS, PMS). - Allow the POS system to pre-select the Payment Entry Mode at the POI, and so avoid manual selection at the POI (PED Device), would this be needed. The only value considered now is "W" and allows the POS/PMS to preselect Staged Digital Wallets (SDW) as the payment method.
CardInvoiceCompanyId	M	A	2	Planet reference of Invoice company (bank): CI, CE, ...
CardInvoiceCompanyName	M	AN	30	Name of invoice company; Citicorp, Cekab, ...
CardIssueNumber	O/M	N	2	Card issue number. Optional in case the card has been swiped.
CardNumber	O/M	N	20	Card PAN. No start and end sentinel.
CardNumberEncrypted	O/M	N	20	Encrypted card PAN. No start and end sentinel.
CardSchemeId	M	A	2	Gives indication on how the payment scheme of the transaction. In 3CXml response messages it indicates the scheme used for the transaction.(VS, AX, MC, ...) In 3CXml request messages it indicates the scheme to be used at the POI for the transaction. Currently it is used only for Staged Digital Wallets (SDW), so in combination with the CardInputmethod set to "W", and values currently defined for the SDW are "AP" (Alipay), "WC" (Wechat), "3W" (3C Wallet). In case the field is not present or empty, the selection of the scheme will be made at

				the POI. In case the value is unknown or the product is not supported by the payment device, the request will be rejected.
CardSchemeLabel	M	AN	30	Name of the card scheme like "Visa", "Mastercard", "Diners", ...
CardSchemeName	M	AN	30	Name of the card scheme like "Visa", "Mastercard", "Diners", ...
CardStartDate	O	N	4	Card start date. Format MMyy.
CardTrack1Data	O	N	40	Card track 1 raw data.
CardTrack2Data	O/M	N	50	Card track 2 raw data. Mandatory in case the card is swiped.
CardTrack3Data	O	N	40	Card track 3 raw data.
COFIndicator	C	AN	1	Gives indication on the nature of the agreement set between the payer (the customer) and the payee (the merchant). The values currently defined for this field are: • "C" for Unscheduled (UCOF)- This value should be used when the customer card credentials are being stored in the customer profile (e.g Card-on-file wallet) • "I" for Instalments • "R" for Recurring • "P" for Industry-Practice MITs (No-Show, Delayed Charge, ...) • "N" for Non-Card-On-File (Credentials are not to be stored)
Currency	O	A	3	Currency of the request. 3 character ISO definition: EUR, USD, ... In case no currency is provided, the local currency is used.
CurrencyUsed	M	AN	3	Currency used in the process. Currency requested or local currency.
CVV2	N/I	AN	4	Card verification value.
CVV2Result	O	AN	1	Result code of the card verification value.
DCCFlag	O	Y/N	1	Flag indicating the transaction is a DCC one or not. Default = N.
DCCMarkup	O	N	10	Markup which is applied to the DCC rate provided by the DCC provider.
EmvApplicationId	M	AN	32	Reference of application use during an EMV transaction
EmvAuthorized	M	Y/N	1	Feedback to know if an EMV authorization has been done or not
EmvCryptogram	M	AN	16	EMV cryptogram reference
EmvCryptogramType	M	AN	2	Cryptogram description
EmvData1	O	AN	40	Any additional EMV data for future use.
EmvData2	O	AN	40	Any additional EMV data for future use.
EmvPinVerified	M	Y/N	1	Feedback to know if the PIN was entered
EmvScenarioId	O	A	2	2 bytes Id to indicate the EMV scenario to choose. Default is DT.
EmvTerminalId	O	AN	25	Reference of EMV terminal used to get EMV card details
EmvTerminalIdManufacturer	O/M	AN	25	Internal reference manufacturer of EMV terminal (e.g. serial number)
EmvTerminalMessage	O	A/M	50	A free message to be displayed on the terminal. Can contain for instance table number, guest reference, ...
EmvTerminalVerifResult	O/M	AN	10	Status of the different functions as seen from the terminal
EmvTrack2Equivalent	O/M	AN	40	EMV track 2
EmvTransactionStatusInformation	O/M	AN	4	Identifies the status of the card at the end of the transaction process
EmvUnpredictableNumber	O/M	AN	8	An unpredictable number is generated for each EMV transaction.
InvoiceId	O/M	AN	50	The reference printed on the invoice or check number on the receipt.
MarketData	O	AN	255	Market specific data. Refer to section "Market Specific Data" for more information on the data tag names.
MerchantId	M	AN	20	Contract number between the merchant and the invoice company.
Message	O	AN	90	Text message which can come from a remote authorization host or from the Integra application.
MITType	C	AN	1	Industry-practice MIT. Values currently defined are: • "N" for No-Show • "L" for Delayed Charges • "S" for Resubmission • "R" for Reauthorization

OriginalRTRN	O	ANS	20	Original RequesterTransRefNum. Merchant Reference of the transaction which the current transaction refers to (e.g. To link a refund to a previous sale).
PosEnvironment	O	A	1	POS Environment indicator. "M" for MOTO, "R" for Recurring, "I" for Installment, "D" for Deferred.
PosVersion	O	ANS	40	Version of the POS/PMS System or application. Used for information only.
PrintData1	O	AN	1000	Any additional text to be printed out on the transaction receipt. The merchant receipt.
PrintData2	O	AN	1000	Any additional text to be printed out on the transaction receipt. The cardholder receipt.
Result	M	A	1	Result of the operation. A (accepted), R (rejected), E (Warning), ...
ResultReason	M	AN	2	Indicates the reason of the result. XB (local bad card type), ...
SCATransRef	C	AN	40	Issuer/acquirer Transaction reference of the initial SCA transaction. To be sent by the merchant for any further MIT related to that initial transaction.
SCAExemptionInd	O	AN	1	In requests and responses. Indicates that the current transaction is Out-of-Scope for PSD2 or that an exemption is requested or was applied, not requiring Strong Customer Authentication. The value provided in the field indicates the type of SCA Exemption or type of PSD2 Out-Of-Scope reason. Current list of defined values are (not all implemented): • "S" for Secure Corporate Payment (Exemption) • "V" for Low-Value Transaction (Exemption) • "W" for Whitelisted beneficiary (Exemption) • "T" for Transaction Risk Analysis (Exemption) • "U" for Unattended (Exemption) • "C" for C-Less Transaction with no CVM (Exemption) • "I" for One-Leg-Out – Issuer (Out-Of-Scope) • "L" for One-Leg-Out – Acquirer (Out-Of-Scope) • "A" for Anonymous Payment (Out-Of-Scope)
TaxFreeData	C	XML		TaxFreeData is a structure of XML data fields, specific to Tax Free processing. This field is conditional as it must be present when Tax Free processing is to be provided as part of the current transaction. More details about the content is provided in the corresponding Annex of this document.
TdsAuthentResultCode	O	AN	20	3DS Visa Transaction Status (TC)
TdsAuthentIndicator	O	AN	20	3DS Visa Electronic Commerce Indicator (ECI), Mastercard Universal Card Authentication Field (UCAF)
TdsAuthentCryptogram	O	AN	255	3DS Visa Cardholder Authentication Verification Value (CAVV), Mastercard Account Authentication Value (AAV)
TdsDSTransID	O	AN	40	Directory Server Transaction ID
TdsTransIdentifier	O	AN	255	3DS Visa Transaction Identifier (XID)
TdsVersion	O	AN	1	3DS version
TimeStamp	M	AN	14	Actual local time of the transaction. Format "yyyyMMddHHmmss". 24 hour formatting. Optional in the reply, depending on solution.
Token	O	N	40	Token, instead of or in addition of card number. SDW: the token returned in 3CXml response messages of SDW payments is not a card token. If the merchants wants to reverse the SDW payment, or refund partially or completely the SDW payment, the Token returned with the initial SDW payment must be submitted by the ECR in the corresponding reversal or refund request.
TransactionUID (with Type)	C	AN	128	New field introduced with support of Staged Digital Wallets (SDW) in our card present solutions. Alphanumeric string representing either a Barcode (Type="BARCODE") or a QR Code (Type="QRCODE"). Size is currently defined on max 128 chars, this is currently enough to support the SDW. This is an optional field, which is used in architectures where the SDW user Barcode or QRCode is read by the ECR and must be transmitted to Planet to proceed

				with the payment. This is a conditional field which can be present only when the CardInputMethod is "W" and a valid SDW CardSchemeId is also communicated.
TransInitiator	C	AN	1	Transaction Initiator, "M" for Merchant, "C" for cardholder.
TransRefNum	O	AN	10	Integra FE internal transaction reference number. Or for EP2 solutions transaction reference coming from the host.
UserData1	O	AN	40	Any data the integrator wants to have saved with the transaction. This field will also be presented on the Planet reconciliation report following each transaction
UserData2	O	AN	40	Any data the integrator wants to have saved with the transaction.
UserData3	O	AN	40	Reserved for future use
UserData4	O	AN	40	Reserved for future use
ZoneId	O/M	AN	20	An identification of the geographical zone (in hotel, restaurant or shop) the request is related to. With this parameter specific EMV terminals are associated to zones.

5.4.7 Variant request fields

Some of the fields are mandatory, optional, or not applicable, depending on the scenario used.

EftSettmementType/Field	*Sale-Terminal*	*Refund-Terminal*	*Sale*	*Refund*	*Completion*	*Sale-Reversal*	*Refund-Reversal*	*Completion-Reversal*
AuthCodeInputMethod	O	O	O	O	M	O	O	O
BankAuthCode	O	O	O	O	M	O	O	O
CardExpiryDate	N/A	N/A	O	O	O	O	O	O
CardInputMethod	N/A	N/A	M	M	O	O	O	O
CardNumber	N/A	N/A	O	O	O	O	O	O
InvoiceId	O	O	N/A	N/A	N/A	N/A	N/A	N/A

5.4.8 Variant reply fields

Some of the reply fields do not exist for all the request types. In general the application returns the EMV fields if possible. This means in case EMV data is available. In case of a fallback the EMV fields will be empty.

5.4.9 Examples

Example of "Sale-Terminal" request type. The transaction request is sent directly to a terminal "Term1". The request has been accepted locally. The result reason is "TO" and there was no authorization code received by the bank.

```
<Request
  Type="EftSettlementEmv"
  SequenceNumber="37625"
  RequesterTransRefNum="078639">
<EftSettlementType>Sale-Terminal
</EftSettlementType>
<EmvScenarioId>DT<EmvScenarioId>
<EmvTerminalId>Term1</EmvTerminalId>
<Amount>20.00</Amount>
<Currency>EUR</Currency>
<TimeStamp>20041019144802</TimeStamp>
</Request>
```

```
<Response
  Type="EftSettlementEmv"
  SequenceNumber="37625"
  RequesterTransRefNum="078639">
<Result>A</Result>
<ResultReason>TO</ResultReason>
<BankResultCode></BankResultCode>
<BankAuthCode></BankAuthCode>
<CardNumber>45234.....22818</CardNumber>
<CardExpiryDate>0411<CardExpiryDate>
<CardTrack2Data>4521.....22818D0411101.....5209667
<CardTrack2Data>
<CardSchemeId>VS</CardSchemeId>
<CardSchemeName>VISA</CardSchemeName>
<CardFunctionId>CC</CardFunctionId>
<CardFunctionName>credit card</CardFunctionName>
<CardInvoiceCompanyId>NT</CardInvoiceCompanyId>
<CardInvoiceCompanyName>49 NATWEST
</CardInvoiceCompanyName>
<Amount>20.00</Amount>
<AmountExtra>0</AmountExtra>
<AmountUsed>20.00</AmountUsed>
<CurrencyUsed>EUR</CurrencyUsed>
<TransRefNum>34543661</TransRefNum>
<EmvPinVerified>Y</EmvPinVerified>
<EmvAuthorized>Y</EmvAuthorized>
<EmvTerminalId>Term1</EmvTerminalId>
<EmvTrack2Equivalent>45234.....22818D0411201.....5209667
</EmvTrack2Equivalent>
<EmvApplicationId>A0000000031010</EmvApplicationId>
<EmvCryptogram>75AE0125E817BA90</EmvCryptogram>
<EmvCryptogramType>10</EmvCryptogramType>
<EmvUnpredictableNumber>606B814B</EmvUnpredictableNu
mber>
</Response>
```

Example of a request where the transaction can be done in any terminal associated to zone id "**rest1**". The waiter will see a message like **"Table 15 (21.00)"** in the transaction option list on the terminal. Instead of choosing out of a transaction list he will also be able to choose the transaction by using the reference "**42144**". This one might be printed on the receipt, or on the invoice. The authorisation has been processed online.

```
<Request
  Type="EftSettlementEmv"
  SequenceNumber="37625"
  RequesterTransRefNum="078639">
<EftSettlementType>Sale-Terminal
</EftSettlementType>
<EmvScenarioId>AL<EmvScenarioId>
<EmvTerminalId></EmvTerminalId>
<EmvTerminalMessage>Table15</EmvTermin
alMessage>
```

```
<Response
  Type="EftSettlementEmv"
  SequenceNumber="37625"
  RequesterTransRefNum="078639">
<Result>A</Result>
<ResultReason>00</ResultReason>
<BankResultCode>00</BankResultCode>
<BankAuthCode>947385</BankAuthCode>
<CardNumber>4521.....22818</CardNumber>
<CardExpiryDate>0411<CardExpiryDate>
```

```
<ZoneId>rest1</ZoneId>
<InvoiceId>42144</InvoiceId>
<Amount>21.00</Amount>
<Currency>GBP</Currency>
<TimeStamp>20041019144802</TimeStamp>
</Request>
```

```
<CardTrack2Data>4521.....22818D0411101.....5209667v
<CardTrack2Data>
<CardSchemeId>VS</CardSchemeId>
<CardSchemeName>VISA</CardSchemeName>
<CardFunctionId>CC</CardFunctionId>
<CardFunctionName>credit card</CardFunctionName>
<CardInvoiceCompanyId>NT</CardInvoiceCompanyId>
<CardInvoiceCompanyName>49 NATWEST
</CardInvoiceCompanyName>
<Amount>20.00</Amount>
<AmountExtra>0</AmountExtra>
<AmountUsed>20.00</AmountUsed>
<CurrencyUsed>GBP</CurrencyUsed>
<TransRefNum>34543661</TransRefNum>
<EmvPinVerified>Y</EmvPinVerified>
<EmvAuthorized>Y</EmvAuthorized>
<EmvTerminalId>Term1</EmvTerminalId>
<EmvTrack2Equivalent>4521.....22818D0411101.....5209667</
EmvTrack2Equivalent>
<EmvApplicationId>A0000000031010</EmvApplicationId>
<EmvCryptogram>75AE0125E817BA90</EmvCryptogram>
<EmvCryptogramType>10</EmvCryptogramType>
<EmvUnpredictableNumber>606B814B</EmvUnpredictableNu
mber>
</Response>
```

Example DCC request where a token is passed in the request and no terminal is used. In the response the masked card number is returned. Here the request was rejected...

```
<Request
  Type="EftSettlementEmv"
  SequenceNumber="128"
  RequesterTransRefNum="131003091623"
  RequesterLocationId="9999931"
  RequesterStationId="ST001">
<EftSettlementType>Completion
</EftSettlementType>
<Token>6049742569087734</Token>
<Amount>1.260</Amount>
<Currency>CHF</Currency>
<DCCFlag>Y</DCCFlag>
<LocalAmount>1.00</LocalAmount>
<LocalCurrency>EUR</LocalCurrency>
<BinRate>1.260</BinRate>
<BinAmount>1.260</BinAmount>
<BinCurrency>CHF</BinCurrency>
<AmountTotal>1.00</AmountTotal>
<AuthCodeInputMethod>A</AuthCodeInputM
ethod>
<BankAuthCode>081633</BankAuthCode>
<BankResultCode>00</BankResultCode>
<TimeStamp>20130930143048</TimeStamp>
<CardTrack2Data></CardTrack2Data>
<UserData1>trans-222333</UserData1>
<UserData2></UserData2>
</Request>
```

```
<Response
  RequesterLocationId="9999931"
  RequesterStationId="ST001"
  RequesterTransRefNum="131003091623-00"
  SequenceNumber="128"
  Type="EftSettlementEmv">
<Amount>1.26</Amount>
<AmountExtra>0.00</AmountExtra>
<AmountUsed>1.26</AmountUsed>
<BankAuthCode>081633</BankAuthCode>
<BankResultCode>96</BankResultCode>
<BinAmount>1.26</BinAmount>
<BinCurrency>CHF</BinCurrency>
<BinRate>1.260765</BinRate>
<CardExpiryDate>1220</CardExpiryDate>
<CardFunctionId>CC</CardFunctionId>
<CardFunctionName>Credit Card</CardFunctionName>
<CardInputMethod>S</CardInputMethod>
<CardInvoiceCompanyId>3C</CardInvoiceCompanyId>
<CardInvoiceCompanyName>Test New
Auth</CardInvoiceCompanyName>
<CardNumber>490118xxxxxx7734</CardNumber>
<CardSchemeId>VS</CardSchemeId>
<CardSchemeName>VISA</CardSchemeName>
<CurrencyUsed>CHF</CurrencyUsed>
<DCCFlag>Y</DCCFlag>
<EftSettlementType>Completion</EftSettlementType>
<LocalAmount>1.0</LocalAmount>
```

<LocalCurrency>EUR</LocalCurrency>
<MerchantId>100009851</MerchantId>
<Message>PROCESS NOT POSSIBLE</Message>
<Result>R</Result>
<ResultReason>96</ResultReason>
<TransRefNum>0634</TransRefNum>
<EmvPinVerified>N</EmvPinVerified>
<BankTerminalId>93C00610</BankTerminalId>
<SignatureRequired>N</SignatureRequired>
<Token>6049742569087734</Token>
</Response>

5.4.10 "EftSettlementEmv" (Sale) non-EMV requests

The **"EftSettlementEmv" (Sale) message can also be used for NON-EMV requests**. Integra would search for EMV data but would not find it and thus do a non-EMV authorisation (if required) and settlement, instead of an EMV one.

5.5 Message "EftData"

5.5.1 Purpose

The EftData message is used to send back all possible Eft data related to a current or former transaction processed by Integra. If the data is still available Integra sends it back. If no data found Integra send back an error message. Integra shall provide the data which is related to the RequesterTransRefNum. If the RequesterTransRefNum is different than the ongoing or last one no data is returned and the result "E" is returned.

5.5.2 Request and reply

<Request
Type="EftData"
SequenceNumber="10 bytes"
RequesterTransRefNum="20 bytes"
[RequesterStationId="10 bytes"]
[RequesterLocationId="10 bytes"]
[ValidationId="64 bytes"]
[ValidationCode="64 bytes"]>
</Request>

<Response
Type=" EftData "
SequenceNumber="10 bytes"
[RequesterTransRefNum="20 bytes"]
[RequesterStationId="10 bytes"]
[RequesterLocationId="10 bytes"]
[ValidationId="64 bytes"]
[ValidationCode="64 bytes"]>
<Result>1</Result>
[<Message>90 bytes</Message>]

[as much data as possible of ongoing request or returned in last message to integrator]

</Response>

The RequesterTransRefNum field is mandatory in the request. This is the reference needed to find back the transaction data.

The response returns the result and the other data fields which were included in the last response sent to the integrator. In case of an ongoing request an empty result is returned.

The **<result>** field in the response will be filled up with the following parameters:

- <empty>: if the request is currently being processed
- A: if the referenced transaction was accepted, (same result field as when the last transaction was done)
- R: if the referenced transaction was rejected, (same result field as when the last transaction was done)
- E: in case the transaction could not be found or the RequesterTransRefNum is not the one of the ongoing or last transaction

5.6 Message "DCCCheck"

5.6.1 Purpose

Certain credit cards are **"DCC capable"**. This means that the cardholder can use his card with the local currency of the merchant, or his own home card currency (BIN currency). The DCCCheck message is used to find out if the card is DCC eligible or not. In case of yes, the BIN currency and rate are returned in the message response.

The DCCCheck message can be sent at any time. But it should be done at least just before the settlement is done, to get the last valid rate of the card.

5.6.2 Request and reply

<Request
Type="DCCCheck"
SequenceNumber="10 bytes"
RequesterTransRefNum="20 bytes"
[RequesterStationId="10 bytes"]
[RequesterLocationId="10 bytes"]
[ValidationId="64 bytes"]
[ValidationCode="64 bytes"]>
<LocalAmount>10 bytes</LocalAmount>
<LocalCurrency>123</LocalCurrency>
<CardInputMethod>12</CardInputMethod>
<CardNumber>20 bytes</CardNumber>
[<Token>40 bytes</Token>]
<CardExpiryDate>MMyy</CardExpiryDate>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
[<CardTrack2Data>50 bytes</CardTrack2Data>]
</Request>

<Response
Type="DCCCheck"
SequenceNumber="10 bytes"
RequesterTransRefNum="20 bytes"
[RequesterStationId="10 bytes"]
[RequesterLocationId="10 bytes"]
[ValidationId="64 bytes"]
[ValidationCode="64 bytes"]>
<Result>1</Result>
<ResultReason>12</ResultReason>
<CardNumber>20 bytes</CardNumber>
[<Token>40 bytes</Token>]
<CardExpiryDate>MMyy</CardExpiryDate>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
<LocalAmount>10 bytes</LocalAmount>
<LocalCurrency>123</LocalCurrency>
<DCCFlag>Y|N</DCCFlag>
[<BinRate>14 bytes</BinRate>]
[<BinAmount>10 bytes</BinAmount>]
[<BinCurrency>123</BinCurrency>]
[<BinCurrencyDescription>30 bytes
</BinCurrencyDescription>]
[<Message>90 bytes</Message>]
[<DCCMarkup>10 bytes</DCCMarkup>]
</Response>

5.6.3 Amount and currency fields

A DCC transaction is a transaction which is not done in the local currency, but in the BIN currency of the card. Like this you could do a USD transaction in Europe for instance. To be able to do this the integrator must have done a **DCCCheck** message to get the BIN currency of the card and the exchange rate related to the local currency. After this a settlement can be done where the integrator provides the different DCC parameters received. The DCC flag in the settlement indicates whether the integrator requests a DCC transaction or not.

Some rules exist on the amount and currency fields.

- In all cases the **<LocalAmout>** field, **if present,** must correspond to the local amount of the Integra application. If not the request is rejected.

- In case the **<DCCFlag>** field is set to **N** (default), the amount and currency of the transaction are taken of the **<Amount>** and **<Currency>** fields.

- In case the **<DCCFlag>** field is set to **Y**, the amount and currency of the transaction are still taken of the **<Amount>** and **<Currency>** fields. The following checks on the fields are done:

  1. <BinAmount> = <Amount>
  2. <BinCurrency> = <Currency>

5.6.4 Fields

Name	Pres	Type	Max len	Content
BinAmount	O/M	N	10	Local amount multiplied by the bin rate. Mandatory if DCCFlag set to Y.
BinCurrency	O/M	A	3	Card's BIN currency. Mandatory if DCCFlag set to Y.
BinCurrencyDescription	O/M	AN	30	Card's BIN currency in text. Mandatory if DCCFlag set to Y.
BinRate	O/M	N	14	Rate of the BIN vs the local currency. Mandatory if DCCFlag set to Y.
CardExpiryDate	M	N	4	Card expiration date. Format MMyy.
CardInputMethod	M	A	2	"E" for EMV chip (contact ICC), "P" for proximity (contactless NFC), "S" for swipe (track), "M" for manual entered.
CardNumber	M	N	20	Card PAN. No start and end sentinel.
CardTrack2Data	O/M	N	50	Card track 2 raw data. Mandatory in case the card is swiped. For the format refer to section **Error! Reference source not found.**.
DCCFlag	M	Y/N	1	Indicates whether the card is DCC capable or not.
DCCMarkup	O	N	10	Markup which is applied to the DCC rate provided by the DCC provider.
LocalAmount	M	N	10	The local (merchant) amount.
LocalCurrency	M	A	3	The local currency.
Message	O	AN	90	Any additional text message (success, error, ...)
Result	M	A	1	Result of the operation. A (accepted), R (rejected), E (Warning), ... Refer to section **Error! Reference source not found.** for the content.
ResultReason	M	AN	2	Indicates the reason of the result. Either local value, or result from remote server. Refer to section **Error! Reference source not found.** for the content.
TimeStamp	M	AN	14	Actual local time of the request. Format "yyyyMMddHHmmss". 24 hour formatting
Token	N/I	N	40	Token, instead of or in addition of card number

Remarks

- The same local validations on the card number than on the CardCheck request are done.
- In case the local currency in the request does not correspond to the local currency of the Integra application the request will be rejected.

5.6.5 Examples

Valid DCC check

```
<Request
   Type="DCCCheck"
   SequenceNumber="4322"
   RequesterTransRefNum="123133">
<LocalAmount>100</LocalAmount>
<LocalCurrency>EUR</LocalCurrency>
<CardInputMethod>M</CardInputMethod>
<CardNumber>41111.....1111</CardNumber>
<CardExpiryDate>0107</CardExpiryDate>

<TimeStamp>20041022173455</TimeStamp>
</Request>
```

```
<Response
   RequesterLocationId=""
   RequesterStationId=""
   RequesterTransRefNum="123133"
   SequenceNumber="4322"
   Type="DCCCheck">
<BinAmount>128.16</BinAmount>
<BinCurrency>USD</BinCurrency>
<BinCurrencyDescription>US Dollar
</BinCurrencyDescription>
<BinRate>1.2816418</BinRate>
<CardExpiryDate>0107</CardExpiryDate>
<CardNumber>41111.....1111</CardNumber>
<DCCFlag>Y</DCCFlag>
<LocalAmount>100</LocalAmount>
<LocalCurrency>EUR</LocalCurrency>
<Message>Success, no error.</Message>
<Result>A</Result>
<ResultReason>00</ResultReason>
<TimeStamp>20041022173455</TimeStamp>
</Response>
```

DCC check for a card not DCC capable.

```
<Request
   Type="DCCCheck"
   SequenceNumber="4322"
   RequesterTransRefNum="123133">
<LocalAmount>100</LocalAmount>
<LocalCurrency>EUR</LocalCurrency>
<CardInputMethod>M</CardInputMethod>
<CardNumber>41111.....1111</CardNumber>
<CardExpiryDate>0107</CardExpiryDate>
<TimeStamp>20041022173455</TimeStamp>
</Request>
```

```
<Response
   RequesterLocationId=""
   RequesterStationId=""
   RequesterTransRefNum="123133"
   SequenceNumber="4322"
   Type="DCCCheck">
<BinAmount />
<BinCurrency />
<BinCurrencyDescription>Not available
</BinCurrencyDescription>
<BinRate />
<CardExpiryDate>0107</CardExpiryDate>
<CardNumber>41111.....1111</CardNumber>
<DCCFlag>N</DCCFlag>
<LocalAmount>100</LocalAmount>
<LocalCurrency>EUR</LocalCurrency>
<Message>No DCC info available for the card
number.</Message>
<Result>R</Result>
<ResultReason>03</ResultReason>
<TimeStamp>20041022173455</TimeStamp>
</Response>
```

The DCC check message is used in swipe card solutions where the integrator reads the card number and wants to check DCC capability before a transaction is made. Or it can also be used for integrators handling themselves the EMV terminal, and want to provide a DCC choice on their terminal.

5.7 Message reply "Error"

In case the application receives a request which cannot be parsed an error message will be sent back to the requester. This can also be done in case there is an internal application error not foreseen. The error code and message field give information about the type of error.

```
<Response Type="Error">
  <ErrorCode>1234</ErrorCode>
  [<Message>90 bytes</Message>]
</Response>
```

5.8 Message "CheckStatus"

5.8.1 Purpose

The CheckStatus message is used to check the readiness of the application. The RequesterTransRefNum attribute of the header has not to be provided in this request.

5.8.2 Request and reply

```
<Request
  Type="CheckStatus"
  SequenceNumber="10 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
</Request>
```

```
<Response
  Type="CheckStatus"
  SequenceNumber="10 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
<Status>12</Status>
[<Message>90 bytes</Message>]
</Response>
```

The **<Status>** field in the reply return the following values:

- **0**: inactive
- **1**: active
- **2**: unknown

The **<Message>** field can contain any additional text.

5.8.3 Example

```
<Request
  Type="CheckStatus"
  SequenceNumber="00127"
  RequesterStationId=" Sta-64"
  RequesterLocationId="362005">
</Request>
```

```
<Response
  RequesterLocationId="362005"
  RequesterStationId="Sta-64"
  SequenceNumber="00127"
  Type="CheckStatus">
<Result>A</Result>
<Status>1</Status>
<Message>Active</Message>
</Response>
```

5.9 Message "ShiftOpen"

This request opens a reconciliation period. The date passed in the request is taken as shift open date.

Note: In case EMV terminals are connected to that location, all the terminals are placed into an "open" (ready) state. Depending on the solution the EftTerminalControl command shiftOpen might be sent to them.

5.9.1 Request and reply

```
<Request
  Type="ShiftOpen"
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
</Request>
```

```
<Response
  Type="ShiftOpen"
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
<ResultReason>123</ResultReason>
<Status>12</Status>
[<Message>90 bytes</Message>]
</Response>
```

In case the shift is already open an error message will be sent back.

5.9.2 Examples

Valid shift open request

```
<Request
  Type="ShiftOpen"
  RequesterLocationId="362005"
  SequenceNumber="1234"
  RequesterStationId="269304">
<TimeStamp>20040518152300</TimeStamp>
</Request>
```

```
<Response
  RequesterLocationId="362005"
  RequesterStationId="269304"
  SequenceNumber="1234"
  Type="ShiftOpen">
<Message>Shift open success - station
active</Message>
<Result>A</Result>
<ResultReason>XO</ResultReason>
<Status>1</Status>
</Response>
```

Shift open request, but shift already open

```
<Request
  Type="ShiftOpen"
  RequesterLocationId="362005"
  SequenceNumber="1234"
  RequesterStationId="269304"
  RequesterTransRefNum="123456">
<TimeStamp>20040518152300</TimeStamp>
</Request>
```

```
<Response
  RequesterLocationId="362005"
  RequesterStationId="269304"
  RequesterTransRefNum="123456"
  SequenceNumber="1234" Type="ShiftOpen">
<Message>Shift open failed - shift already
opened</Message>
<Result>R</Result>
<ResultReason>XR</ResultReason>
```

```
<Status>1</Status>
</Response>
```

5.10 Message "ShiftClose"

This request closes a reconciliation (booking) period. The date passed in the request is taken as shift close date.

Note: In case EMV terminals are connected to that location, all the terminals are placed into a "closed" (not ready) state. Depending on the solution the EftTerminalControl commands shiftClose and reconciliation might be sent to them.

5.10.1 Request and reply

```
<Request
  Type="ShiftClose"
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
</Request>
```

```
<Response
  Type="ShiftClose"
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
<ResultReason>123</ResultReason>
<Status>12</Status>
[<Message>90 bytes</Message>]
[<PrintData1>1000 bytes</PrintData1>]
</Response>
```

In case the shift is already open an error message will be sent back.

5.10.2 Example

```
<Request
  Type="ShiftClose"
  RequesterLocationId="362005"
  SequenceNumber="1234"
  RequesterStationId="269304">
<TimeStamp>2004051rrr8152300</TimeStamp>
</Request>
```

```
<Response
  RequesterLocationId="362005"
  RequesterStationId="269304"
  SequenceNumber="1234"
  Type="ShiftClose">
<Message>Shift close failure - bad date time
parameter received</Message>
<Result>E</Result>
<ResultReason>XE</ResultReason>
<Status>1</Status>
</Response>
```

Note: a receipt might have to be generated at close shift. In this case the formatted receipt data is sent back in the PrintData1 field.

5.11 Message "ShiftCloseOpen"

This is a request for a close, then for an open reconciliation action. It sends back the same error message than the separate close and open requests. In case the shift is already closed the shift is just opened.

Note: In case EMV terminals are connected to that location, all the terminals are closed and then opened again. Depending on the solution the EftTerminalControl commands shiftClose and reconciliation might be sent to them.

5.11.1 Request and reply

```
<Request
  Type="ShiftCloseOpen"
  SequenceNumber="10 bytes"
  RequesterTransRefNum="20 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
</Request>
```

```
<Response
  Type="ShiftCloseOpen"
  SequenceNumber="10 bytes"
  RequesterTransRefNum="20 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
<ResultReason>123</ResultReason>
<Status>12</Status>
[<Message>90 bytes</Message>]
[<PrintData1>1000 bytes</PrintData1>]
</Response>
```

Note: a receipt might have to be generated at close shift. In this case the formatted receipt data is sent back in the PrintData1 field.

5.12 Message "Cancel"

In the current implementation in case a request is made and the requester disconnects from the Integra application the request is going to be cancelled, and possibly reversed if a transactions had been created.

In addition to a disconnect ion an extra "Cancel" message exists in the protocol to cancel the request previously done. Like this the integrator may cancel a request and still stay connected to the Integra application.

> Note, it is not always possible to cancel a transaction. If e.g. an online Sale request has been sent to the acquirer, a transaction has been created and it is too late to cancel, and the request will be rejected. In case a cancel is rejected and the transaction should ve voided a reversal request shall be sent.

5.12.1 Request and reply

```
<Request
  Type="Cancel"
  SequenceNumber="10 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<SequenceNumberToCancel>10 bytes
<SequenceNumberToCancel>
</Request>
```

```
<Response
  Type="Cancel"
  SequenceNumber="10 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
[<Message>90 bytes</Message>]
</Response>
```

The **<SequenceNumberToCancel>** must contain the value of the SequenceNumber attribute of the original request to cancel.

The **<Result>** field in the reply will be filled up with the following parameters:

- A: accepted
- R: rejected, as not possible, not implemented or any other reason
- E: in case the sequence number to cancel is not found

The **<message>** field can contain any additional text.

5.12.2 Example

```
<Request Type="Cancel" SequenceNumber="456"
 <SequenceNumberToCancel>455
 <SequenceNumberToCancel>
</Request>
```

```
<Response Type="Cancel" SequenceNumber="456"
 <Result>A</Result>
 <Message>OK</Message>
</Response>
```

5.13 Message "GetToken" (Bulk)

5.13.1 Purpose

The **GetToken** message is used to tokenize a number (n) of card PAN in one single message exchange (Request/Response). This is particularly useful in situations where no card validation, authorization or payment are needed, just a tokenization. Typically such method is used when migrating card information from one provider to another.

The number of cards (n) to tokenize is between 1 and 50.

5.13.2 Request and Reply

```
<Request
  Type="GetToken"
  SequenceNumber="10 bytes"
  RequesterTransRefNum="20 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<CardData>
  <SeqNum>12</SeqNum>
  <CardNumber>20 bytes</CardNumber>
  <CardExpiryDate>MMyy</CardExpiryDate>
</CardData>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
</Request>
```

```
<Response
  Type="GetToken"
  SequenceNumber="10 bytes"
  RequesterTransRefNum="20 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
<ResultReason>12</ResultReason>
<CardData>
  <SeqNum>12</SeqNum>
  <Token>40 bytes</Token>
  <CardNumber>20 bytes</CardNumber>
  <CardSchemeId>2 bytes</CardSchemeId>
  [<CardSchemeName>30 bytes</CardSchemeName>]
</CardData>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
</Response>
```

5.13.3 Fields

Name	Pres	Type	Max len	Content
CardData	M			Data Structure (Container). There can be up to "n" (0<n<51) CardData fields (Containers) included in the message, so maximum 50.
CardNumber	M	AN	20	PAN of the card. This is the masked number in the response message.
Token	M	AN	40	In response messages. Card Token returned by 3C.
CardExpiryDate	M	AN	4	Card Expiration date in MMYY format.
CardSchemeId	M	AN	2	2-chars value as defined in the 3C systems, e.g. "VS, "MC, ...
CardSchemeName	O	ANS	30	Name of the Card Scheme as define in our systems, e.g. "Visa", "Mastercard", ...Returned when available/defined for the Card Scheme
TimeStamp	M	AN	14	Actual local time of the request. Format "yyyyMMddHHmmss". 24 hour formatting

5.13.4 Examples

```
<Request
   Type="GetToken"
   SequenceNumber="123456"
   RequesterTransRefNum="RTRN123456789"
   RequesterStationId="MyStation"
   RequesterLocationId="654321"
<CardData>
   <SeqNum>1</SeqNum>
   <CardNumber>4242424242424242</CardNumber>
   <CardExpiryDate>1220</CardExpiryDate>
</CardData>
<CardData>
   <SeqNum>2</SeqNum>
   <CardNumber>5454545454545454</CardNumber>
   <CardExpiryDate>0321</CardExpiryDate>
</CardData>
<TimeStamp>20200924175233</TimeStamp>
</Request>
```

```
<Response
   Type="GetToken"
   SequenceNumber="123456"
   RequesterTransRefNum=" RTRN123456789"
   RequesterStationId="MyStation"
   RequesterLocationId="654321"
<Result>A</Result>
<ResultReason>XO</ResultReason>
<CardData>
   <SeqNum>1</SeqNum>
   <Token>1358476534671254549</Token>
   <CardNumber>424242******4242</CardNumber>
   <CardSchemeId>VS</CardSchemeId>
   <CardSchemeName>VISA</CardSchemeName>
</CardData>
<CardData>
   <SeqNum>2</SeqNum>
   <Token>5695003774555578941</Token>
   <CardNumber>545454******5454</CardNumber>
   <CardSchemeId>MC</CardSchemeId>

<CardSchemeName>MASTERCARD</CardSchemeNa
me>
</CardData>
<TimeStamp>20200924175233</TimeStamp>
</Response>
```

5.14 Message "GetPAN" (Bulk)

5.14.1 Purpose

The GetPAN message is used to de-tokenize a number (n) of card tokens in one single message exchange (Request/Response). This is particularly useful in situations where an integrator needs to retrieve the card data which was used for tokenisation. Typically such method is used when migrating card information from one provider to another.

The number of cards (n) to tokenize is between 1 and 50.

5.14.2 Request and Reply

```
<Request
  Type="GetPAN"
  SequenceNumber="10 bytes"
  RequesterTransRefNum="20 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<CardData>
  <SeqNum>12</SeqNum>
  <Token>40 bytes</Token>
</CardData>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
</Request>
```

```
<Response
  Type="GetPAN"
  SequenceNumber="10 bytes"
  RequesterTransRefNum="20 bytes"
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
<ResultReason>12</ResultReason>
<CardData>
  <SeqNum>12</SeqNum>
  <CardNumber>20 bytes</CardNumber>
  <CardExpiryDate>MMyy</CardExpiryDate>
  <CardSchemeId>2 bytes</CardSchemeId>
  [<CardSchemeName>30 bytes</CardSchemeName>]
</CardData>
<TimeStamp>yyyyMMddHHmmss</TimeStamp>
</Response>
```

5.14.3 Fields

Name	Pres	Type	Max len	Content
CardData	M			Data Structure (Container). There can be up to "n" (0<n<51) CardData fields (Containers) included in the message, so maximum 50.
CardNumber	M	AN	20	PAN of the card. This is the clear PAN in the response message.
Token	M	AN	40	In request messages. Card Token previously returned by 3C.
CardExpiryDate	M	AN	4	Card Expiration date in MMYY format.
CardSchemeId	M	AN	2	2-chars value as defined in the 3C systems, e.g. "VS, "MC, ...
CardSchemeName	O	ANS	30	Name of the Card Scheme as define in our systems, e.g. "Visa", "Mastercard", ...Returned when available/defined for the Card Scheme
TimeStamp	M	AN	14	Actual local time of the request. Format "yyyyMMddHHmmss". 24 hour formatting

5.14.4 Examples

```
<Request
  Type="GetPAN"
  SequenceNumber="123456"
  RequesterTransRefNum="RTRN123456789"
  RequesterStationId="MyStation"
  RequesterLocationId="654321"
<CardData>
  <SeqNum>1</SeqNum>
  <Token>1358476534671254549</Token>
</CardData>
<CardData>
  <SeqNum>2</SeqNum>
  <Token>5695003774555578941</Token>
</CardData>
<TimeStamp>20200924175822</TimeStamp>
</Request>
```

```
<Response
  Type="GetPAN"
  SequenceNumber="123456"
  RequesterTransRefNum=" RTRN123456789"
  RequesterStationId="MyStation"
  RequesterLocationId="654321"
<Result>A</Result>
<ResultReason>XO</ResultReason>
<CardData>
  <SeqNum>1</SeqNum>
  <CardNumber>4242424242424242</CardNumber>
  <CardSchemeId>VS</CardSchemeId>
  <CardSchemeName>VISA</CardSchemeName>
</CardData>
<CardData>
  <SeqNum>2</SeqNum>
  <CardNumber>5454545454545454</CardNumber>
  <CardSchemeId>MC</CardSchemeId>

<CardSchemeName>MASTERCARD</CardSchemeNa
me>
</CardData>
<TimeStamp>20200924175822</TimeStamp>
</Response>
```

5.15 Message "EftTerminalStatus"

5.15.1 Purpose

The EftTerminalStatus is a message which is used to either retrieve certain status information from the terminal, or if configured, automatically sends back status messages to the integrator. It can be used by integrators who need more information about the status of the terminal, than just the transaction response reply. The following status information might be available, depending on the terminal solution.

- Display
- Card
- Power
- Error
- Info

If Integra is configured to automatically send back status message responses the integrator does not need to send a status request. Message responses are send back on the same communication channel than on the one a transaction request is made. Automatic status messages are only send back in the lifetime of an ongoing transaction, this means between the Eft message request and EFT message response.

5.15.2 Request and reply

```
<Request
  Type=" EftTerminalStatus"
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<StatusType>1</StatusType>
<EmvTerminalId>25 bytes</EmvTerminalId>
[<TimeStamp>yyyyMMddHHmmss</TimeStamp>]
[<Data>1000 bytes</Data>]
</Request>
```

```
<Response
  Type=" EftTerminalStatus"
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<StatusType>1</StatusType>
<EmvTerminalId>25 bytes</EmvTerminalId>
[<StatusMessage>90 bytes</StatusMessage>]
<StatusCode>4 bytes</StatusCode>
[<Data>1000 bytes</Data>]
[<Data1>1000 bytes</Data1>]
[<Data2>1000 bytes</Data2>]
[<CardNumber>20 bytes</CardNumber>]
[<CardExpiryDate>MMyy</CardExpiryDate>]
[<CardTrack1Data>40 bytes</CardTrack1Data>]
[<CardTrack2Data>50 bytes</CardTrack2Data>]
[<CardTrack3Data>40 bytes</CardTrack3Data>]
[<Token>40 bytes</Token>]
</Response>
```

5.15.3 Fields

Name	Pres	Type	Max len	Content
EmvTerminalId	M	N	25	The terminal the status info is requested for.
StatusType	M	A	1	Status type which is requested, or provided in the response.
StatusCode	M	N	4	Unique status code
StatusMessage	O	A	90	A text message related to the status code sent back
Data	O	AN	1000	Extra data passed in the request or back in the response. Usage depends on message status type.
Data1	O	AN	1000	Additional data sent back in the response. Depends on terminal integration.
Data2	O	AN	1000	Additional data sent back in the response. Depends on terminal integration.
CardNumber	O	N	40	Card number, if required for integrator. The card number might be masked.
CardExpiryDate	O	N	4	Card expiry data, if required for integrator
TimeStamp	O	AN	14	Actual local time of the request. Format "yyyyMMddHHmmss". - 24 hour formatting
Token	O	N	40	Token, instead of or in addition of card number
CardTrack1Data	O	N	40	Card track 1 raw data.
CardTrack2Data	O/M	N	50	Card track 2 raw data. Mandatory in case the card is swiped. For the format refer to section Error! Reference source not found..
CardTrack3Data	O	N	40	Card track 3 raw data.

5.15.4 Status types

The following status types are mandatory in the request, or provided in the response to the integrator.

5.15.4.1 "D": terminal display status, including the text displayed

The response message provides the state of the terminal display. The StatusCode provided is one of the following ones, if the content of the text is known. In case the content is not defined the status code 125 will be sent. These messages can be displayed "as is" on the integrator display.

Status Code	Name	Comment
100	STATUS_DISPLAY_INSERT_SWIPE	enter card number manually
101	STATUS_DISPLAY_INSERT	
102	STATUS_DISPLAY_SWIPE	
103	STATUS_DISPLAY_MANUAL_PAN	
104	STATUS_DISPLAY_MANUAL_EXP_DATE	enter expiry date manually
105	STATUS_DISPLAY_MANUAL_ISSUE	enter issue number manually
106	STATUS_DISPLAY_ENTER_PIN	
107	STATUS_DISPLAY_APPROVED	
108	STATUS_DISPLAY_DECLINED	
109	STATUS_DISPLAY_REFERRAL	
110	STATUS_DISPLAY_ONLINE_AUTH	
111	STATUS_DISPLAY_DCC_CHECK	
112	STATUS_DISPLAY_DCC_SELECTION	
113	STATUS_DISPLAY_CARD_NOT_ACCEPTED	
114	STATUS_DISPLAY_SIGNATURE_CHECK	
115	STATUS_DISPLAY_GRATUITY_QUESTION	
116	STATUS_DISPLAY_GRATUITY_ENTRY	
117	STATUS_DISPLAY_GRATUITY_SUMMARY	
118	STATUS_DISPLAY_CONFIRMATION_AMOUNT	
119	STATUS_DISPLAY_WELCOME_MESSAGE	
120	STATUS_DISPLAY_AMOUNT_ENTRY	Amount entry in case of standalone
121	STATUS_DISPLAY_REMOVE_CARD	the card should be removed from the reader
122	STATUS_DISPLAY_SELECT_DEBIT_CREDIT	displayed when terminal is waiting for user selection
123	STATUS_DISPLAY_IDLE	
124	STATUS_DISPLAY_TRANSACTION_CANCELLED	
125	**STATUS_DISPLAY_UNKNOWN**	**Any display text received from the terminal. Possibly translated into terminal language.**
126	STATUS_DISPLAY_CONTACT_ACQUIRER	
127	STATUS_DISPLAY_TRANS_INFO	

Important note:

The <Data1> field contains additional information about displaying the messages on the integrator displays, in case no display exists on the terminal. In this case the string value "external" is included, the integrator system should try to display the status message in the external cardholder screen (e.g. for unattended solution) and if possible/existing in the merchant display.

This function is used e.g. for parking systems where the messages on the entry station are displayed in the parking screen instead of in the terminal screen, and where it is mandatory to display some messages. Like for instance the card application choice or card removal screen in case of error. If not displayed the cardholder could stand in front of the terminal not knowing what to do in specific situations.

5.15.4.2 "C": terminal card insertion status, or card data, if configured

The "C" type provides codes about the status of the card. For code 205 it should provide the card number and expiry date in the corresponding fields.

Status Code	Name	Comment
200	STATUS_CARD_INSERTED	Card has been inserted.
201	STATUS_CARD_REMOVED	Card has been removed.
202	STATUS_CARD_SWIPED	A card has been swiped.
203	STATUS_CARD_NOT_PRESENT	
204	STATUS_CARD_MANUALLY_ENTERED	
205	STATUS_CARD_DATA_ACCEPTED	The card has been accepted by Integra. Card number, and/or token, and/or expiry date provided.
206	STATUS_CARD_DATA_NOT_ACCEPTED	The card has NOT been accepted by Integra. Card number, and/or token, and/or expiry date provided.
207	STATUS_CARD_DATA_ERROR	Error while reading the card. The card could not be read.
208	STATUS_CARD_CANCELED	The cardholder presses the CNL key when the terminal is asking for card insertion.
209	STATUS_CARD_TIMEOUT	When something goes wrong and nothing happen during long period when waiting for card insertion.

5.15.4.3 "P": terminal power status

The "P" type provides information about the power status of the terminal.

Status Code	Name	Comment
300	STATUS_POWER_ON	
301	STATUS_POWER_OFF	

5.15.4.4 "I": other information

The "I" type can provide specific terminal information, such as diagnostic data, or initial receipt data. Not supported for all solutions.

Status Code	Name	Comment
400	STATUS_TERMINAL_INFO	General terminal information. Some additional data might be provided in the response <data> tag, such as: ▪ SoftwareVersion ▪ ServiceRequestsTime ▪ TerminalIp ▪ TerminalIpMask ▪ TerminalIpGw ▪ TerminalIpDns
401	STATUS_CHANGE_LANGUAGE	Change of language, either automatic due to language of chip card, or by selection of user. The UserData1 field contains the ISO 639-1 language code minus the country code. The message field contains some explicative text (in English).

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<Request Type="EftTerminalStatus" SequenceNumber="14" RequesterLocationId="LU6CDNLP01"
RequesterTransRefNum="20140128-000">
 <EmvScenarioId>DT</EmvScenarioId>
 <EmvTerminalId>POS1</EmvTerminalId>
 <StatusType>I</StatusType>
 <TimeStamp>20140203101108</TimeStamp>
</Request>

<Response
Type="EftTerminalStatus"
SequenceNumber=""
RequesterTransRefNum="">
<EmvTerminalId>POS1</EmvTerminalId>
<EmvTerminalRefManufacturer>99990055</EmvTerminalRefManufacturer>
<StatusType>I</StatusType>
<StatusCode>400</StatusCode>
<TimeStamp>20140203065131424</TimeStamp>
<Data>
<SoftwareVersion>ccterminal-1.0.0-140130-b-tac</SoftwareVersion>
<ServiceRequestsTime>20140204004800</ServiceRequestsTime>
<Language>en-GB</Language>
<TerminalIp>192.168.003.182</TerminalIp>
<TerminalIpMask>255.255.255.000</TerminalIpMask>
<TerminalIpGw>192.168.003.001</TerminalIpGw>
<TerminalIpDns>192.168.003.010</TerminalIpDns>
</Data>
</Response>
```

5.15.5 Examples

The example shows the status messages after the integrator has sent an EFT request, and the user inserts the card.

```
<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="0908040001"
SequenceNumber="0002" Type="EftTerminalStatus">
<EmvTerminalId>POS1</EmvTerminalId>
<StatusType>D</StatusType>
<StatusMessage>Display Status: idle</StatusMessage>
<StatusCode>123</StatusCode>
<Data2>COMMUNICATION</Data2>
</Response>

<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="0908040001"
SequenceNumber="0004"
Type="EftTerminalStatus">
<EmvTerminalId>POS1</EmvTerminalId>
<StatusType>D</StatusType>
<StatusMessage>Display Status: Insert card</StatusMessage>
<StatusCode>101</StatusCode>
<Data2>COMMUNICATION</Data2>
</Response>

<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="0908040001"
SequenceNumber="0004"
Type="EftTerminalStatus">
<EmvTerminalId>POS1</EmvTerminalId>
<StatusType>D</StatusType>
<StatusMessage>Insert card</StatusMessage>
<StatusCode>125</StatusCode>
<Data1>display</Data1>
<Data2>COMMUNICATION</Data2>
</Response>

<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="0908040001"
SequenceNumber="0005"
Type="EftTerminalStatus">
<EmvTerminalId>POS1</EmvTerminalId>
<StatusType>C</StatusType>
<StatusMessage>Card Status: inserted</StatusMessage>
<StatusCode>200</StatusCode>
<Data2>COMMUNICATION</Data2>
</Response>
...
```

In the example above the message "Insert card" should be displayed on the cardholder screen of the parking system in case a terminal without screen is used.

5.16 Message "EftTerminalControl"

5.16.1 Purpose

The EftTerminalControl message is used to execute specific operations (commands) on dedicated terminals. Operations such as close or open the shift of the terminal, activate, deactivate, release card and possible others. Not all the commands are implemented for all terminals handled by Integra.

5.16.2 Request and reply

```
<Request
  Type="EftTerminalControl"
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Command>40 bytes</Command>
<EmvTerminalId>25 bytes</EmvTerminalId>
[<Data>40 bytes</Data>]
[<AttendantId>20 bytes</AttendantId>]
</Request>
```

```
<Response
  Type=" EftTerminalControl "
  SequenceNumber="10 bytes"
  [RequesterTransRefNum="20 bytes"]
  [RequesterStationId="10 bytes"]
  [RequesterLocationId="10 bytes"]
  [ValidationId="64 bytes"]
  [ValidationCode="64 bytes"]>
<Result>1</Result>
[<Message>90 bytes</Message>]
[<Data1>40 bytes</Data1>]
[<Data2>40 bytes</Data2>]
[<Data3>40 bytes</Data3>]
[<PrintData1>1000 bytes</PrintData1>]
[<PrintData2>1000 bytes</PrintData2>]
[<Mac>40 bytes</Mac>]
</Response>
```

The response returns the result as always and an optional message.

The **<result>** field in the response will be filled up with the following parameters:

- A: if accepted
- R: if rejected
- E: in case of an error

The **request** can have the following **<Command>** values as specified in the next sections. The **<EmvTerminalId>** field specifies the terminal to be controlled, and is mandatory. The **<Data>** field is optional and depends on the command used. Refer to the command description for more information. The fields **<PrintData1>** and **<PrintData2>** might contain receipt information, e.g. in case a shift is closed on a terminal.

The following commands are currently supported:

- **activate**: activate a terminal, the terminal becomes operational
- **deactivate**: deactivate a terminal, and the terminal becomes non-operational, i.e. the card reader and the pinpad cannot be used
- **shiftOpen**: opens the shift of a terminal
- **shiftClose**: closes the shift of a terminal
- **closeOpenShift**: closes, then opens the shift of a terminal
- **getEmvSettings**: get the EMV settings of EMV cards defined in Integra
- **getBinRanges**: get the bin ranges of cards supported by Integra
- **notify**: notification status changes on integrator side

- **initialize**: initialize the EMV terminal with some parameters, and trigger a configuration and SW download
- **getKeys**: provide security keys (for PIN, data encryption, MAC generation) to be injected into an EMV terminal
- **reconciliation**: this operation requests the current booking period counter from the EFT. It does not modify the state of the booking period
- **reconciliationWithClosure**: performs a booking period close operation and returns the resulting booking period counter

The requirement of sending these messages to Integra FE or the terminal depends on the solution. By default if Integra FE communicates to the integrator it handles these operations towards the terminal. Only in case the integrator needs more control of the terminal for specific reasons the integrator can use them.

Also, depending on the terminal solutions not all commands are supported.

Always check with our project engineering to determine what are the detailed terminal actions behind a command, as it might vary from terminal to terminal.

5.16.3 Command "activate"

5.16.3.1 Purpose

This request can be used to activate an EMV terminal. Activate will place the terminal in a "technical" operational state so it can be used by the integrator to execute operations. This command is only required in case the integrator has formerly deactivated the terminal. No online message is sent to the acquirer host.

5.16.3.2 Request and reply

```
<Request
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Command>activate</Command>
   <EmvTerminalId>POS1</EmvTerminalId>
</Request>
```

```
<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Result>A</Result>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Command>activate</Command>
</Response>
```

Note: <Data> field parameter

In case the field **<Data>** contains the value "**disableCardCheckOnline**" no online call is made when the card is inserted into the terminal. This is to configure a specific handling for the situation where a terminal provides the 3cXml protocol and normally when the card is inserted an online call towards an Integra server is made to check the card and receive a token.

5.16.4 Command "deactivate"

5.16.4.1 Purpose

This request can be used to deactivate an EMV terminal. Deactivate will place the terminal into a "technical" non-operational state so it can NOT be used by the integrator to execute operations, and deactivates the reader and pinpad. This command is only required in case the integrator explicitly wants to deactivate a terminal to be sure no operations can be done. **No online message is sent to the acquirer host.**

Note: <Data> field parameter

The field **<Data>** should contain the value "**doorOpen**" in case the door is opened for a parking system. In some countries and for some terminal solutions this will trigger a closing of the terminal.

As an example it is required for parking solutions in Scandinavia whenever the pay station door is opened.

5.16.4.2 Example request and reply

```
<Request
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Command>deactivate</Command>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Data> </Data>
</Request>
```

```
<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Result>A</Result>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Command>deactivate</Command>
</Response>
```

```
<Request
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Command>deactivate</Command>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Data>doorOpen </Data>
</Request>
```

```
<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Result>A</Result>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Command>deactivate</Command>
</Response>
```

5.16.5 Command "shiftOpen"

5.16.5.1 Purpose

This request can be used to open the shift of a specific an EMV terminal. It shall be sent if the terminal has previously been closed by a shiftClose command. The terminal might also send an online message to the Integra Front-End host. Verify with Planet if this message is required.

5.16.5.2 Example request and reply

```
<Request
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Command>shiftOpen</Command>
   <EmvTerminalId>POS1</EmvTerminalId>
   <AttendantId>Peter</AttendantId>
</Request>
```

```
<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Result>A</Result>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Command>shiftOpen</Command>
   <PrintData1>....</PrintData1>
</Response>
```

This command shall only be used in combination with shiftClose command. Unless there are specific reasons it is not required to use the command, as Integra handles the shift of the terminals by default.

In case a synchronization of the integrator and Integra shift is required it should be rather considered to use the general ShiftOpen message, which handles the shift all terminals at the same time (no specific terminal ID is required in the request).

The **PrintData1** field might contain some totals report of the transactions done for this shift. This is the case for EP2 terminals.

For EP2 terminals the **AttendantId** field shall contain the identification of the attendant, which is used for all transactions of this shift, and send back in the PrintData1 field (report) of the shiftClose command of that shift.

5.16.6 Command "shiftClose"

5.16.6.1 Purpose

This request can be used to close the shift of a specific an EMV terminal. In case the terminal is directly communicating with the acquirer all remaining transactions on the terminal shall be uploaded before the terminal is closed. As for the deactivate command the terminal will be placed out of operational state. The terminal might also send an online message to the Integra Front-End host. Verify with Planet if this message is required.

5.16.6.2 Example request and reply

```
<Request
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Command>shiftClose</Command>
   <EmvTerminalId>POS1</EmvTerminalId>
</Request>
```

```
<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Result>A</Result>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Command>shiftClose</Command>
   <PrintData1>....</PrintData1>
</Response>
```

This command shall only be used in combination with shiftOpen. Unless there are specific reasons it is not required to use the command, as Integra handles the shift of the terminals by default.

In case a synchronization of the integrator and Integra shift is required it should be rather considered to use the general ShiftClose message, which handles the shift all terminals at the same time (no specific terminal ID is required in the request).

5.16.7 Command "closeOpenShift"

5.16.7.1 Purpose

This request can be used to first close, then right after open a shift for a specific EMV terminal. For simplicity to not use two messages. This is the normal use case of the message.

For solutions where the terminal is acting as a standalone connected as client and processing payments requests to Integra and the message In this case, Integra sends back a string to indicate a whether new EMV settings have to be retrieved by the terminal.

5.16.7.2 Example request and reply

```
<Request
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <EmvTerminalId>POS1</EmvTerminalId>
   <Command>closeOpenShift</Command>
</Request>
```

```
<Response
RequesterLocationId="332200"
RequesterStationId="2"
RequesterTransRefNum="1006280010"
SequenceNumber="0001"
Type="EftTerminalControl">
   <Command>closeOpenShift</Command>
   <EmvTerminalId>POS1</EmvTerminalId>
   <Result>A</Result>
   <Message></Message>
   <Data1>UpdateEmvSettings<Data1>
   <PrintData1>....</PrintData1>
</Response>
```

In the response sent by Integra, if the field **<Data1>** contains the string "**UpdateEmvSettings**, this indicates that the terminal has to update its EMV settings and bin ranges.

- If "Data1" is set to empty the terminal has nothing to do.
- If "Data1" contains the string "UpdateEmvSettings, the terminal has to request new EMV settings and bin ranges to Integra by sending multiple "EftTerminalControl/getEmvSettings" commands and one "EftTerminalControl/getBinRanges" command.

Note: a receipt might have to be generated at close shift. In this case the formatted receipt data is sent back in the PrintData1 field.

!?!

In case the integrator connects to an Integra central host, the **commands** shiftOpen, shiftClose and closeOpenShift commands must not be used in all cases. In general the main **messages** ShiftOpen, ShiftClose and ShiftCloseOpen are sufficient. These messages close or open ALL the terminals linked to the location at the same time.

BUT there might be specific situations where it might be necessary to independently close one specific terminal, for instance before replacement and to make sure it uploads transactions or execute some action in a menu. Closing a terminal might involve uploading of transactions for terminals directly communicating with banks.

The **PrintData1** field might contain some totals report of the transactions done for this shift.

5.16.8 Command "initialize"

5.16.8.1 Purpose

This message command can be send to initialize an EMV terminal with some parameters from the request, and to trigger a configuration and software download.

The following parameters can be set:

- The terminal Id
- The language of the terminal
- The terminal's time
- Network, to set DHCP, DNS, network mask, gateway address and IP address of terminal
- Flag to disable receipt generation in reply

The message is also used to trigger a software and configuration update

This message is only implemented for some EMV terminal types. Check with our project engineering team if the message is available for the terminal used in the solution.

The **EmvTerminalId** is the terminal ID of the terminal. Value to be provided by Planet, as configured in the Integra-FE.

5.16.8.2 Data field values

For the initialize command the Data field contains a number of XML sub-tags.

XML Data field	Description
DisableReceipt	Flag set to generate the receipt in the reply, or to not generate it. Merchant and cardholder receipts in the fields PrintData1 and PrintData2 can be set independently, with the following values: • "A": disable all receipts • "M": disable merchant receipt • "C": disable cardholder receipt
Language	Format: "language-Country" If you do not want to set the language just do not send the tag "Language" at all.
TerminalIpDhcp	"Y" to be set to DHCP, "N" to static IP address
TerminalIp	Terminal IP address, if static IP
TerminalIpMask	Network mask
TerminalIpGw	Terminal's network gateway server address
TerminalIpDns	Terminal's network DNS server address
TimeStamp	Time of the terminal with the format "yyyyMMddHHmmss". If you do not want to set the time just do not send the tag "TimeStamp" at all. The time is local time.
TmsServerIp1	The terminal's terminal management system main IP address
TmsServerIp2	The terminal's terminal management system backup IP address

Notes:

- **Network setup.** To set DHCP, DNS, network mask, gateway address and IP address of terminal. If one of these parameters is missing when DHCP is not used then the request will be rejected. If you do not want to configure the network then do not send any of the following tags in the request: TerminalIpDhcp, TerminalIp, TerminalIpMask, TerminalIpGw, TerminalIpDns

- **Software/Config update.** To trigger a Software/Config update. If this is wanted the following fields must be filled: EmvTerminalId, TmsServerIp1, TmsServerPort1. Each value for each parameter will be provided by Planet. If you do not want to trigger a Software/Config update then just do not send the tag EmvTerminalId, TmsServerIp1 and TmsServerPort1 in the request.

5.16.8.3 Example request and reply

```
<Request
SequenceNumber="1"
RequesterLocationId="LU6CDNLP05"
RequesterTransRefNum="20140106-000"
Type="EftTerminalControl" >
<EmvTerminalId>99990055-
30381740</EmvTerminalId>
<Command>initialize</Command>
<Data>
<Language>en-US</Language>
<TimeStamp>20140106154955</TimeStamp>
<TerminalIpDhcp>N</TerminalIpDhcp>
<TerminalIp>192.168.003.182</TerminalIp>
<TerminalIpMask>255.255.255.000</TerminalIpMask>
<TerminalIpGw>192.168.003.001</TerminalIpGw>
<TerminalIpDns>192.168.003.010</TerminalIpDns>
<TmsServerIp1>153.46.253.5</TmsServerIp1>
<TmsServerPort1>8953</TmsServerPort1>
</Data>
</Request>
```

```
<Response
 Type="EftTerminalControl"
 SequenceNumber="1"
 RequesterTransRefNum="20140106-000">
<Command>initialize</Command>
<Result>A</Result>
<Data>
<SoftwareVersion>ccterminal-
1.0.0</SoftwareVersion>
<ServiceRequestsTime>20140107011400</ServiceReq
uestsTime>
<Language>en-GB</Language>
<TerminalIp>192.168.3.182</TerminalIp>
<TerminalIpMask>255.255.255.0</TerminalIpMask>
<TerminalIpGw>192.168.3.1</TerminalIpGw>
<TerminalIpDns>192.168.003.010</TerminalIpDns>
</Data>
</Response>
```

6 Communication methods

6.1 Introduction

This chapter provides some information about the datalink and transport layers implemented in the Integra application, to accept requests by using the 3cXml protocol. The interface partner only sees the result of the combination protocol/datalink/transport, which is the communication method. The following sections describe how the communication is established, and give an example. As example a card CardCheck is given. The interface partner can actually choose between one of the provided connection methods.

6.2 Raw TCP/IP socket

This is the most basic way to communicate, but also the recommended one, as the communication is most under control. The Integra application provides a server TCP/IP port where the client application can connect to. The client opens the connection and sends the raw XML request as it. The request is handled, and a response is send back to the client. After the response has been sent back the client can to close the connection or stay connected. In case the client disconnects before the response is sent back the request is cancelled by the Integra application.

Example:

```
<?xml version="1.0" ?>
<Request
        Type="CardCheck"
        SequenceNumber="127"
        RequesterStationId="12"
        RequesterTransRefNum="5">

        <CardInputMethod>S</CardInputMethod>
        <CardNumber>411111……1111</CardNumber>
        <CardExpiryDate>0106</CardExpiryDate>
        <CardTrack2Data>411111……1111D0404101……000000</CardTrack2Data>
</Request>

<?xml version="1.0" ?>
<Response
        Type="CardCheck"
        SequenceNumber="127"
        RequesterStationId="12"
        RequesterTransRefNum="5">

        <Result>A</Result>
        <ResultReason>XO</ResultReason>
        <CardNumber>41111……11111</CardNumber>
        <CardExpiryDate>0106</CardExpiryDate>
        <CardSchemeId>VS</CardSchemeId>
        <CardSchemeName>Visa</CardSchemeName>
        <CardFunctionId>C</CardFunctionId>
        <CardFunctionName>Credit</CardFunctionName>
        <CardInvoiceCompanyId>NT</CardInvoiceCompanyId>
        <CardInvoiceCompanyName>Natwest</CardInvoiceCompanyName>
        <Message>Accepted with floor limit</Message>
```

```
</Response>
```

Remarks:

- A maximum number of connections are allowed during the same time.
- This communication is not secure. Data exchange is visible on the network.

6.3 SSL socket

Instead of using raw TCP/IP SSL can be used. Standard Public key infrastructure (PKI) certificates are used to in the SSL handshakes. Openssl can be used to test the SSL.

It is possible to use server (Integra) authentication only, or server (Integra) and client (Integrator) authentication. SIX will provide a server certificate to use.

In order to comply with current PCI PA-DSS regulations at minimum SSL version 3 or TLS version 1 must be used, and one of the following ciphers:

- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_DSS_WITH_AES_128_CBC_SHA
- TLS_KRB5_WITH_RC4_128_MD5
- TLS_KRB5_WITH_RC4_128_SHA
- TLS_KRB5_WITH_3DES_EDE_CBC_MD5
- TLS_KRB5_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_WITH_RC4_128_MD5
- SSL_RSA_WITH_RC4_128_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA
- SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA

Remark: these parameters might change in the future, as PCI might become more restrictive. Contact Us to get an updated list of protocols and ciphers supported.

6.4 HTTP or HTTPS

In this communication mode also TCP/IP is used, but the XML requests and responses are included in an HTTP or HTTPS POST request. The Integra HTTP server listens to a port specified (e.g. port 80, 5000) for incoming requests. In case a request is done, it is processed and an HTTP response is returned. In case of processed requests (successful or not), a 200 "OK" reply is returned. In case a request is not recognized, or an internal error happens, a 500 "internal error" reply is sent back. After the reply is sent back the TCP/IP connection is closed. In case the client disconnects before the response is sent, the request is canceled by the Integra application.

Important: Additional parameters in POST request header and url.

For larger installations with a hosted central server (clustered Integra to cope with the amount of transactions per day) to provide the required SLA a load balancer is in front of the farm and routing the requests towards different Integra servers. To be able to optimize the routing and have some kind of flexibility when it comes to the configuration it is requested to have some additional parameters in the POST url and header.

1. We need the fields RequesterLocationId and RequesterStationId to be present in the request header.

2. We need the **RequesterLocationId in the url as REST parameter.** The url shall start with the string serviceIntegra then use the string ***requesterLocationId*** to identify the requester location id.

Example:

```
POST https://194.7.101.47:11400/serviceIntegra/requesterLocationId/604001 HTTP/1.0
Charset=utf-8
Content-Type: text/xml
RequesterLocationId: 604001
RequesterStationId: 12
Content-Length: 32

<?xml version="1.0" ?>
<Request
Type="CardCheckEmv"
SequenceNumber="127"
RequesterLocationId="604001"
RequesterStationId="12"
RequesterTransRefNum="5">
<Type>Check</Type>
<CardInputMethod>S</CardInputMethod>
<CardNumber>41111……11111</CardNumber>
<CardExpiryDate>0106</CardExpiryDate>
<CardTrack2Data>411111…..11111D0404101…..0000000</CardTrack2Data>
</Request>

HTTP/1.0 200 OK
Server: cccintegra
Date: Friday, 26-Mar-04 15:35:01 GMT
Last-modified: Friday, 26-Mar-04 15:35:01 GMT
Content-type: text/xml
Content-length: 6372

<?xml version="1.0" ?>
<Response
Type="CardCheckEmv"
SequenceNumber="127"
RequesterStationId="12"
RequesterTransRefNum="5">
<Type>Check</Type>
<Result>A</Result>
<ResultReason>XO</ResultReason>
<CardNumber>41111……11111</CardNumber>
<CardExpiryDate>0106</CardExpiryDate>
<CardSchemeId>VS</CardSchemeId>
<CardSchemeName>Visa</CardSchemeName>
<CardFunctionId>C</CardFunctionId>
<CardFunctionName>Credit</CardFunctionName>
<CardInvoiceCompanyId>NT</CardInvoiceCompanyId>
<CardInvoiceCompanyName>Natwest</CardInvoiceCompanyName>
<Message>Accepted with floor limit</Message>
</Response>
```

Note, when using HTTP the communication is not secure. Data exchange can be seen on the network. No card data shall be transferred over the network. A token solution shall be used. If card data is transferred, to make the communication secure HTTPS has to be used. In this case the server authentication happens after the connection and the Integra server public key certificate is exchanged to encrypt the data transferred over the network.

6.5 Batch files

Integra is able to process payment requestbatch files. The following file format has been defined. Contact our Engineering team for further details about how to transfer the files.

6.5.1 XML format

The XML format starts with a root node "BatchFile". Into this node standard settlement request nodes are embedded, with the same syntax as if they would be send in via HTTPS.

```
<BatchFile

Type="EftBatch"
FileRefNum="40 bytes "
FileName="40 bytes"
FileRequestCount="40 bytes"
FileSequenceNumber="40 bytes"
FileTimeStamp="yyyyMMddHHmmss">

    <Request Type="EftSettlementEmv" SequenceNumber="128" RequesterTransRefNum="43241313"
    RequesterLocationId="604001" RequesterStationId="ST001">
    <EftSettlementType>Completion</EftSettlementType>
    ...
    </Request>
    <Request Type="EftSettlementEmv" SequenceNumber="129" RequesterTransRefNum="432413207"
    RequesterLocationId="604366" RequesterStationId="ST001">
    <EftSettlementType>Completion</EftSettlementType>
    ...
    </Request>
    ...

<BatchFile>
```

6.5.2 BatchFile tag attributes

Name	Pres	Type	Max len	Content
Type	M	A	40	Only type "EftBatch" is currently supported.
FileRefNum	M	AN	40	A unique reference of the file exiting in Greenway.
FileName	M	AN	40	The name of the file, as stored in the file system.
FileRequestCount	M	AN	40	The amount of requests in the file.
FileSequenceNumber	M	N	40	A counter of the file exiting in Greenway.
FileTimeStamp	M	AN	12	The date and time at the creation of the file. Format "yyyyMMddHHmmss". 24 hour formatting

6.5.3 Examples

```
<BatchFile Type="EftBatch" FileRefNum="342134112" FileName="batch001.xml" FileRequestCount="2"
FileSequenceNumber="33421" FileTimeStamp="128">

        <!-- non DCC transaction request -->

        <Request Type="EftSettlementEmv" SequenceNumber="128" RequesterTransRefNum="43241313"
        RequesterLocationId="604001" RequesterStationId="ST001">
                <EftSettlementType>Completion</EftSettlementType>
                <Token>6049741011617734</Token>
                <Amount>450.00</Amount>
                <Currency>EUR</Currency>
                <AmountTotal>450.00</AmountTotal>
                <AuthCodeInputMethod>A</AuthCodeInputMethod>
                <BankAuthCode>075252</BankAuthCode>
                <TimeStamp>20130930170257</TimeStamp>
                <UserData1>12345678901234567890</UserData1>
                <UserData2>12345678901234567890</UserData2>
                <UserData3>12345678901234567890</UserData3>
                <UserData4>12345678901234567890</UserData4>
                <InvoiceId>12345678901234567890</InvoiceId>
        </Request>

        <!-- DCC transaction request -->
        <Request Type="EftSettlementEmv" SequenceNumber="129" RequesterTransRefNum="432413207"
        RequesterLocationId="604366" RequesterStationId="ST001">
                <EftSettlementType>Completion</EftSettlementType>
                <Token>437741011644553</Token>
                <Amount>1.240</Amount>
                <Currency>CHF</Currency>
                <DCCFlag>Y</DCCFlag>
                <LocalAmount>1.00</LocalAmount>
                <LocalCurrency>EUR</LocalCurrency>
                <BinRate>1.240</BinRate>
                <BinAmount>1.240</BinAmount>
                <BinCurrency>CHF</BinCurrency>
                <AmountTotal>1.00</AmountTotal>
                <AuthCodeInputMethod>A</AuthCodeInputMethod>
                <BankAuthCode>111352</BankAuthCode>
                <TimeStamp>20130930170257</TimeStamp>
                <InvoiceId>12345678901234567890</InvoiceId>
                <UserData1>12345678901234567890</UserData1>
                <UserData2>12345678901234567890</UserData2>
                <UserData3>12345678901234567890</UserData3>
                <UserData4>12345678901234567890</UserData4>
        </Request>

</BatchFile>
```

Note: the abbreviated 3cXml protocol could be used to reduce the file size.

```
<BatchFile Type="EftBatch" FileRefNum="342134112" FileName="batch001.xml" FileRequestCount="2"
FileSequenceNumber="33421" FileTimeStamp="128">

        <!-- non DCC transaction request -->
        <Request Type="ESE" SN="128" RTRN="43241313" RLI="604001" RSI="ST001">
```

```
            <EST>CP
            <TK>6049741011617734
            </AM>450.00
            </CUR>EUR
            </AT>450.00
            </ACIM>A
            </BAC>075252
            </TS>20130930170257
            </UD1>12345678901234567890
            </UD2>12345678901234567890
            </UD3>12345678901234567890
            </UD4>12345678901234567890
            </II>12345678901234567890
        </Request>

        <!-- non DCC transaction request -->
        <Request Type="ESE" SN="129" RTRN="432413207" RLI="604366" RSI="ST001">
            </EST>CP
            </TK>437741011644553
            </AM>1.240
            </CUR>CHF
            </DCCF>Y
            </LA>1.00
            </LC>EUR
            </BR>1.240
            </BinAM>1.240
            </BC>CHF
            </AT>1.00
            </ACIM>A
            </BAC>111352
            </BRC>00
            </TS>20130930170257
            </II>12345678901234567890
            </UD1>12345678901234567890
            </UD2>12345678901234567890
            </UD3>12345678901234567890
            </UD4>12345678901234567890
        </Request>

</BatchFile>
```

7 Appendixes

7.1 Market Data Tags

The current chapter contains the list of market data elements (Tags names) to be used in the <MarketData> <Data> subfields. Since the data elements to exchange in the MarketData structure are specific to the industry and therefore vary from one industry to the other, a list of data elements is defined for each industry.

In order to identify to which industry (Car Rental, Lodging, Airline, ...) the list of enclosed data elements refer to, a MarketData attribute named "Type" is defined. The MarketData types are defined in the below table.

In the MarketData structure, each data element is represented by a tag. The Tag length is industry dependant and each tag has a meaning only for the industry type in which it is defined. This also means that the same Tag could theoretically be used in different industry types but would have a different meaning.

7.1.1 Market Data Types

The below list shows the MarketData "Type" values currently allocated to the industries. This list can be extended in the future as well as the data elements contained in the different types.

Industry	MarketData Type
Car Rental	10
Lodging	11
Restaurant	12
Retail	13
Airline	14
Travel/Cruise Industry	15

Even when no data tags are provided the MarketData tag and its type should be present, in order to clearly identify the industry/market type of the system which sends the 3cXml request.

7.1.2 Car Rental Market Data tags

The Car Rental data elements are required only for settlement messages, not for authorization messages. In the Car Rental MarketData structure, each data element is represented by a 4-chars tag. Amounts are provided with implied decimals as indicated in the below table.

Car Rental Industry (Type 10)						
Tag	Format	Max length	Format string	Field Name/Description	M/O	
					Airplus	Amex
AA00	AN	17		Accounting Unit (Customer's)	O	N/A
AD00	AN	17		Department/Business Unit (Customer's)	O	N/A
AJ00	AN	17		Action Number (Customer's)	O	N/A
B500	AN	17		Order Number (Customer's)	O	N/A
BB00	AN	30		Vehicle Type	M	N/A

BN00	D	8	YYYYMMDD	Departure Date (Customer's)	O	N/A
BP00	AN	15		Voucher Number	O	N/A
BS00	N	7	99999V99	Fuel consumption	O	N/A
CN00	N	9	9999999V99	VAT amount	M	N/A
CQ00	N	9	9999999V99	Net amount	M	N/A
DE00	AN	40		CardHolder name	M	N/A
E100	AN	17		Department (Customer's)	O	N/A
E200	AN	17		Discount account	O	N/A
E600	AN	11		Document number	M	N/A
EC00	N	9	9999999V99	Price per Km	O	N/A
EI00	AN	28		The driver of the vehicle	M	M
FL00	N	7	9999999	Driven Km	M	O
G100	AN	17		Internal Account (Customer's)	O	N/A
G500	AN	15		Licence plates	M	O
G600	AN	4		Vehicle Category	M	O
G700	AN	1		Sales indicator	M	N/A
GM00	AN	2		Vehicle Group charged for (Cost Category Code)	O	N/A
GN00	AN	17		Cost Center (Customer's)	O	N/A
GQ00	AN	17		Customer number	O	N/A
GS00	AN	20		Customer Reference	O	N/A
GX00	AN	18		Rate/Service Description	O	N/A
HQ00	AN	2		Mile/KM Indicator	M	O
HX00	D	6	YYMMDD	Check Out Date	M	M
HY00	D	4	HH24MI	Check Out Time	M	O
HZ00	D	6	YYMMDD	Check In Date	M	M
I100	D	4	HH24MI	Check In Time	M	O
I300	N	3	999	Number of days rented	M	N/A
I400	AN	12		Rental Agreement No.	M	M
IW00	AN	17		Employee Number (Customer's)	O	N/A
J900	AN	17		Project Number (Customer's)	O	N/A
JR00	D	8	YYYYMMDD	Invoice Date	O	N/A
JT00	AN	15		Invoice number	O	N/A
JZ00	AN	17		Final Destination	O	N/A
KH00	N	7	9999999	Number of kilometres when returned	M	N/A
KJ00	AN	20		Check In Station	M	O
LK00	AN	3		Rental Rate	O	N/A
LS00	N	7	9999999V99	Telephone units	O	N/A
MA00	N	7	9999999	Kilometers on Pick-Up	M	N/A
MB00	AN	20		Check Out Station	M	O
MG00	AN	25		UIN (VAT Identification Number)	O	N/A
MO00	A	1		Car Rental Company ID	M	N/A
NQ00	AN	30		Point Of Sale Location	O	N/A
OI00	AN	17		Merchant's Order Number	O	N/A
QM00	AN	40		Comments	O	N/A
QR00	AN	27		Miscelleanous Services	O	N/A
VX00	AN	2		Check Out Country	M	M
AX07	AN	18		Check Out (PickUp) City Name	M	M
AX10	AN	3		Check Out (PickUp) Region Code	M	M

AX14	AN	18		Check In (Return) City Name	M	M
AX16	AN	3		Check In (PickUp) Region Code	M	M
AX17	AN	2		Check In (Return) Country Code	M	M
AX24	AN	1		Audit Adjustement Indicator	O	O
AX26	N	9	9999999V99	Audit Adjustement Amount (Same Currency as Transaction)	O	O
AX29	AN	20		Driver Identification Number	O	O
AX30	AN	20		Driver Tax Number	O	O
Group Additional information records (4 minimum per transaction)					**M**	**N/A**
C900	N	9	9999999V99	Collection/ pick-up charges		
C901	N	9	9999999V99	Position's net amount		
C902	N	9	9999999V99	Position's VAT amount		
C903	N	4	99V99	VAT rate		
CB0-0-1-2-3	N	9	9999999V99	Fuel costs		
CC0-0-1-2-3	N	9	9999999V99	Gross amount		
CG0-0-1-2-3	N	9	9999999V99	Down Payment/Deposit		
CI0-0-1-2-3	N	9	9999999V99	Liability insurance amount		
CJ0-0-1-2-3	N	9	9999999V99	Passenger liability insurance		
CT0-0-1-2-3	N	9	9999999V99	Return/ drop-off charges		
CU0-0-1-2-3	N	9	9999999V99	Insurance deductible		
CV0-0-1-2-3	N	9	9999999V99	Misc. deductions with VAT		
CW0-0-1-2-3	N	9	9999999V99	Misc. deductions without VAT		
CX0-0-1-2-3	N	9	9999999V99	Misc. with VAT		
CY0-0-1-2-3	N	9	9999999V99	Misc. without VAT		
D40-0-1-2-3	N	9	9999999V99	Telephone charges		
D70-0-1-2-3	N	9	9999999V99	Insurance package		
D80-0-1-2-3	N	9	9999999V99	Theft Amount		
D90-0-1-2-3	N	9	9999999V99	Full coverage insurance		
DA0-0-1-2-3	N	9	9999999V99	Delivery charges		
O60-0-1-2-3	N	9	9999999V99	Discount		
OQ0-0-1-2-3	N	9	9999999V99	Service/ airport charges		

Example:

```
<MarketData Type="10">
  <Data tag="BN00">20131115</Data>
  <Data tag="KJ00">5687954</Data>
  <Data tag="HX00">131115</Data>
  <Data tag="CC00">12100</Data>
  <Data tag="CC01">10000</Data>
  <Data tag="CC02">2100</Data>
  <Data tag="CC03">21</Data>
  <Data tag="HK00">2500</Data>
  <Data tag="EI00">John Hunter</Data>
</MarketData>
```

7.1.3 Lodging Market Data tags

MarketData for the lodging industry are mostly used for the U.S. market and allows the merchant to benefit from better interchange fees or rates. In the Lodging MarketData structure, each data element is represented by a 2-chars tag. Amounts are provided in the merchant transaction currency with implied decimals as indicated in the below table.

Lodging Industry (Type 11)					
Tag	Format	Max length	Format string	Field Name/Description	M/O
TT	AN	2		Transaction Type	M/M*
CI	N	8	YYYYMMDD	Check In Date	M
CO	N	8	YYYYMMDD	Check Out Date	M
FN	AN	25		Folio Number	M
PT	AN	12		Property Telephone Number	O*
CT	AN	12		Customer Service Telephone Number	O*
R1	N	9	9999999V99	Room Rate 1	O**
N1	N	2		No. of nights at Room Rate 1	O
R2	N	9	9999999V99	Room Rate 2	O
N2	N	2		No. of nights at Room Rate 2	O
R3	N	9	9999999V99	Room Rate 3	O
N3	N	2		No. of nights at Room Rate 3	O
RT	N	9	9999999V99	Room Tax	O**
RN	AN	25		Renter Name	O
GN	AN	25		Guest Name	O
SD	N	2		Stay Duration	M*
PC	N	9	9999999V99	Phone Charges	O
RC	N	9	9999999V99	Restaurant Charges	O
MC	N	9	9999999V99	Mini-Bar Charges	O
SC	N	9	9999999V99	Gift Shop Charges	O
DC	N	9	99999V99	Laundry / Dry Cleaning Charges	O
OI	AN	3		Other Charges Code	O
OC	N	9	9999999V99	Other Charges	O
BA	N	9	9999999V99	Late Charges	O

TR	N	9	9999999V99	Total Room Charges	O
TN	N	9	9999999V99	Total Non-room Charges	O
TC	N	9	9999999V99	Transport Charges	O
GC	N	9	9999999V99	Gratuity Charges	O
CC	N	9	9999999V99	Conference Room Charges	O
VC	N	9	9999999V99	Audio / Visual Charges	O
BC	N	9	9999999V99	Banquet Charges	O
IC	N	9	9999999V99	Internet Charges	O
EC	N	9	9999999V99	Early Departure Charges	O
RS	N	9	9999999V99	Room Service Charges	O
LC	N	9	9999999V99	Lounge / Bar Charges	O
HC	N	9	9999999V99	Health Club Charges	O
PC	N	9	9999999V99	Valet Parking Charges	O
XC	N	9	9999999V99	Cash Disbursement Charges	O
PP	AN	1		Prestigious Property Indicator	O
D0	AN	12		Promotional Code	O
D1	AN	12		Additional Coupon	O
D2	AN	1		Smoking Preference (Y/N)	O
D3	N	2		Number of Rooms Booked by the customer	O
D4	N	2		Number of Adults staying in the room	O
D5	AN	2		Program Code	O
D6	AN	12		Corporate Client Code	O
D7	AN	10		Room Location	O
D8	AN	10		Tax Elements	O
D9	AN	12		Bed Type	O
I0	AN	12		Room Type	O
I1	AN	8		Travel Agency Code	O
I2	AN	25		Travel Agency name	O
I3	AN	12		Rate Type	O

M = Mandatory for Settlement messages
M* = Mandatory for Authorization messages
O = Optional
O* = Required in Settlement messages for reduced interchanges
O** = Required in Settlement messages for MC Corp T+E Rate 3

Additional field explanations

- **Transaction Type:** this field is used to identify the lodging program code of the transaction. The below table shows the defined values and their descriptions.

Value	Description
01	Lodging (Default)
02	No Show reservation
03	Advance Deposit
04	Late Charge

- **Number of nights:** this data tag must be present with a value different than zero if the corresponding room rate data tag is present.
- **Renter Name:** Name of the person or business entity charged for the transaction. It can be different than the guest name.
- **Other Charges Code:** Indicator to explain the amount that is in the "Other charges" field. Value defined by merchant.
- **Prestigious Property Indicator:** this indicator is used by participants to Visa's Prestigious Lodging Program.

Value	Description
"Space"	Default or non-participating property
D	Prestigious property with $500 Limit
B	Prestigious property with $1000 Limit
S	Prestigious property with $1500 Limit

- **Promotional Code:** code identifies current discount program or package.
- **Additional Coupon:** additional coupon or discounts.
- **Program Code:** used to identify special circumstances applicable to the transaction or Cardholder, e.g., frequent guest, upgrade, no-show.
- **Corporate Client Code:** Corporate code assigned to identify the discount rates.
- **Room Location:** Location of the room within the facility, e.g. Park or Garden etc.
- **Tax Elements:** Summary of all tax, e.g. Hotel , tourist.
- **Bed Type:** Size of bed, e.g., king, queen, double.
- **Room Type:** Describes the type of room, e.g., standard, deluxe, suite.
- **Travel Agency:** Code of travel agency that made the lodging facility reservation
- **Rate Type:** Identifies different types of rates, e.g., AAA, Corporate, and Senior Citizen

Example:

```
<MarketData Type="11">
  <Data tag="TT">01</Data>
  <Data tag="CI">20140520</Data>
  <Data tag="CO">20140521</Data>
  <Data tag="FN">123456100</Data>
  <Data tag="SD">1</Data>
  <Data tag="R1">12500</Data>
  <Data tag="N1">1</Data>
  <Data tag="PT">691777888</Data>
</MarketData>
```

7.1.4 Restaurant (Food & Beverage) Market Data tags

In the Restaurant MarketData structure, each data element is represented by a 2-chars tag. Amounts are provided in the merchant transaction currency with implied decimals as indicated in the below table.

Restaurant Industry (Type 12)					
Tag	Format	Max length	Format string	Field Name/Description	M/O
FB	N	9	9999999V99	Food&Beverage amount	O
FA	N	9	9999999V99	Food amount	O
BA	N	9	9999999V99	Beverage Amount	O
T1	N	9	9999999V99	TIP Amount 1	O
E1	AN	20		Employee ID or Name to receive TIP1	O
T2	N	9	9999999V99	TIP Amount 2	O
E2	AN	20		Employee ID or Name to receive TIP2	O
T3	N	9	9999999V99	TIP Amount 3	O
E3	AN	20		Employee ID or Name to receive TIP3	O

Example:

```
<MarketData Type="12">
  <Data tag="FB">14350</Data>
  <Data tag="T1">1000</Data>
  <Data tag="E1">707</Data>
</MarketData>
```

7.2 Tax Free Services – Tax Free Structure of Data (TaxFreeData)

The new 3CXml-ECR Tax Free specific data fields are contained in a specific Node/Data Structure named **<TaxFreeData>**. The presence of the TaxFreeData structure in the 3CXml-ECR message is optional if no Tax Free service/operation is to be managed within the payment flow. When Tax Free services/Operations are expected within the payment process, the fields indicated as mandatory in the <TaxFreeData> structure must be present. Conditional fields need to be present if the requirement criteria are met. These criteria are usually country specific but can also depend on the Tax Free feature/service.

When the required Tax Free data is not present in the request or not valid for the expected Tax Free Service or operation, the payment will be processed regardless and a relevant code and message will be returned in the Tax free structure (<TaxFreeData> field) of the response message.

The processing of any Tax Free service by the Payment Terminal will be executed only if the Payment Terminal is properly configured and enabled accordingly. When a Tax Free service is enabled on the terminal but no TaxFreeData is provided by the POS, the service is bypassed by the terminal, and only the Payment transaction will be processed. When no Tax Free service is enabled on the terminal, and TaxFreeData is provide by the POS, the TaxFreeData is not considered and only the Payment request is processed.

7.2.1 Tax Free Data structure in 3cXml Request (POS to Payment Gateway)

Field name	Cond.	Type	Max Len.	Description
<TaxFreeData>	C	Struct.		Tax Free Data structure. Must be present if any of the Tax Free data fields contained in the structure is present.
<Transaction>	O	Struct.		
<ResponseUrl>	O	ANS		Reserved for future use (web responsive solution)
<FiscalNumber>	O	ANS		Fiscal number to be assigned to the Tax Free form. Mandatory when a form is generated and printed with the payment solution.
</Transaction>	C			
<Receipts>	M	Struct.		Structure to contain receipts data
<Receipt>	M	Struct.		Structure to contain data related to one specific receipt
<UniqueId> *	M	ANS		Unique identifier of the receipt
<Date> *	M	Date		Date of receipt in format yyyy-mm-dd
<Time> *	M	Time		Time of receipt in format hh:mm:ss
<SaleNumber> *	M	ANS		Sale number
<ReceiptNumber>	C	ANS		Fiscal receipt number ("scontrino elettronico")
<TotalItems> *	M	N		Total number of items in the receipt
<Hash> *1	C	ANS		Receipt hash. **Mandatory for Portugal.**
<LineItems>	M	Struct.		Structure to contain line items data
<LineItem>	M	Struct.		Structure to contain data related to one specific line item
<Id> *	M	ANS		Order id of the item
<Description> *	M	ANS		Item description
<VatRate> *	M	ANS		VAT rate in format 0.00
<UnitGrossAmount>	M	ANS		Total line item gross price (includes VAT) in format 0.00
<UnitNetAmount> *	M	ANS		Total line item net price (excludes VAT) in format 0.00
<Quantity> *	M	N		Item quantity
<Eligible> *	M			Eligibility for Tax Free, if false excludes the item
<UnitOfMeasure> *2	C	ANS		**Mandatory only for Poland and Russia.** Description of unit of measure, as per table below:

				PL: sztuka, metr, egzemplarz, komplet, opakowanie, skrzynia, kg, ryza RU: штука, унция, метр, шт, па, комп, уп, короб, м, кг, г
<CategoryCode> *3	C	ANS		**Mandatory only for Spain, Portugal, France.** Product category, as per table below: ES / PT / FR / Category ALI / ALI / ALI / Food and beverage ALT / ALT / ALT / Alcohol and tobacco CUL / CUL / ART / Cultural goods, handcraft DEP / DEP / SPT / Sports and leisure EDM / EDM / APP / Home appliances HOG / CAS / DEC / Home and decoration ELI / PCS / INF / Computing and electronics MOD / MOD / MOD / Fashion and accessories PER / PER / COS / Perfumery, cosmetics and ph JOY / JOI / BIJ / Jewellery and watches DTO / Discounts
<SerialNo> *4	C	ANS		**Mandatory for Portugal if item value > 500 Eur.** Item's serial number: **always include if present even if not mandatory. If not available, populate with the SKU or article code.**
<ArticleCode>	O	ANS		SKU/identifier of the product
</LineItem>				
</LineItems>				
<PaymentMethods>	C	Struct.		**Mandatory for France. To be sent by the POS for the payment methods managed directly at the POS (e.g. Cash, Check).** Optional otherwise.
<PaymentMethod> *5	C	Struct.		**Mandatory for France. To be sent by the POS for the payment methods managed by the POS (e.g. Cash, Check).** Structure to contain payment methods data
<PaymentMethod>	C	ANS		Payment method: **1 – Cash** **2 – Check** 4 – Credit card 13 – Other 14 – Online payment
<Amount>	M	ANS		Amount of the payment in format 0.00
</PaymentMethod>				
</PaymentMethods>				
</Receipt>				
</Receipts>				
<Shopper>	O	Struct.		Structure to contain shopper data
<FirstName>	O	ANS		Shopper's first name
<LastName>	O	ANS		Shopper's last name
<Address>	O	ANS		Shopper's address
<City>	O	ANS		Shopper's city
<Province>	O	ANS		Shopper's city province
<PostCode>	O	ANS		Shopper's address post code
<Country> *6	O	ANS		Shopper's country of residence (ISO 3166-1 numeric. Better to provide this data for accurate Tax Free eligibility result.

<DocumentType>	O	ANS		Document type: IC – Identity card PA – Passport
<DocumentNumber>	O	ANS		Document number
<DocumentExpiryDate>	O	ANS		Document expiry in format yyyy-mm-dd
<Nationality>	O	ANS		Shopper's nationality (ISO 3166-1 numeric)
<DateOfBirth>	O	ANS		Shopper's date of birth in format yyyy-mm-dd
<FiscalCode>	O	ANS		Shopper's fiscal code
<Email>	O	ANS		Email
<MobileNo>	O	ANS		Mobile number with international prefix
</Shopper>				
</TaxFreeData>				

* Mandatory field.
*1 Required only for Portugal.
*2 Required only for Poland and Russia.
*3 Required only for Spain, Portugal and France.
*4 Always include if available, even if not mandatory. For Portugal, this is mandatory for any item with price >= 500 EUR; if not available, populate with SKU or article code.
*5 At least one instance of this node required for France.
*6 Not mandatory, but strongly advised if Fast/immediate refund service has to be offered.

7.2.2 Tax Free Data structure in 3cXml Response (Payment Gateway to POS)

Following the processing of the payment and of the Tax Free service or operation, a response message is returned to the POS, as usual, regardless of the result of these operations. The result/status of the payment and corresponding data are returned in the standard 3CXml fields.

When Tax Free services/operations were processed or attempted the result/status of the operation and corresponding data are returned in the Tax Free structure <TaxFreeData>, with fields described in the below table.

Field name	Cond.	Type	Max Len.	Description
<TaxFreeData>	C	Struct.		Tax Free Data structure. Must be present if any of the Tax Free data fields contained in the structure is present.
<Result>	M	ANS	1	Result of the TaxFree Service ("A", "R", "W", "E")
<ResultReason>	O	ANS	3	List of values to indicate the result of the Tax Free operation or the error.
<Message>	O	ANS	90	Text message to describe the result of the Tax Free operation or the error.
<Eligibility>	O	Struct.		Present if any of the fields contained in the structure is present.
<PreStampRefund>	O	ANS	1	Boolean value that identifies if existing pre-stamp refund options (e.g. Fast Refund) are supported by processed card
<StandardRefund>	O	ANS	1	Boolean value that identifies if standard refund on credit card (PI's refund option code = 0100) is supported by processed card
<Country>	O	ANS	1	Boolean value that identifies if card's country of issuing or shopper's country (if sent in the request) is eligible for Tax Free shopping.

</Eligibility>				
<RefundOptionCode>	O	ANS	4	Code identifying the Refund option selected by the customer on the payment terminal: • 0000 if fast or immediate refund has not been accepted or if required guarantee preauth failed • 1600 if FAST refund has been accepted by the customer • Xxxx if IMMEDIATE refund has been accepted (Value TBD)
<RefundAmount>	O	ANS		Calculated refund amount in format 0.00 (For the option selected)
<PreauthTrxId>	O	ANS		Transaction id representing guarantee preauth for immediate refund processed on terminal (returned with RefundOptionCode=xxxx)
<PreAuthToken>	O	ANS		When the Token generated for the Guarantee (PreAuth) is different from the Token generated for the payment, it is returned in this field.
</TaxFreeData>				

The payment gateway/terminal will return TaxFreeData according to the Tax Free service performed and the result.
In case the shopper refuses immediate refund or if the guarantee pre-authorisation fails, the POS will allow selection of another refund method through the existing process, also based on eligibility flags returned in response.
Beside the TaxFree Data fields, other standard Payment fields will be needed or useful to the POS for completion of the Tax Free process.

- Payment Method Data fields: mainly **<Token>** and **<CardSchemeId>** are relevant as this information is needed in case the payment method used with the payment gateway/terminal is also to be used for the Tax Free refund. In this case the **<Token>** should be transmitted to the Tax Free application (*CreateVoucher* request).
- Payment Receipts: when the Planet payment terminal is used for receipts printing, both the Payment receipt and the Guarantee (PreAuth) receipt are printed by the terminal. In scenarios/deployments where the POS is in charge of printing the receipts (e.g. to include with the invoice), bothe he payment receipt and the Guarantee (PreAuth) receipt must be printed by the POS. The preformatted receipts are then returned by the payment terminal/gateway in the standard 3CXml fields designed for this purpose, **<PrintData1>** and **<PrintData2>**.

7.3 PCI Data Security

7.3.1 Introduction

The Payment Card Industry (PCI) Data Security Standard is the result of the collaboration between VISA and MasterCard and defines guidelines and requirements for any card processing system. The standard encompasses the following areas:

- Build and Maintain a Secure Network
- Protect Cardholder Data
- Maintain a Vulnerability Management Program
- Implement Strong Access Control Measures
- Regularly Monitor and Test Networks
- Maintain an Information Security Policy

Further information regarding PCI can be found at http://pcisecuritystandards.org/

7.3.2 PCI and the Integra 3cXml protocol

The Integra solution is a certified PCI certified solution in general. All components hosted in the Planet data centres (DCs) are subject to a PCI-DSS certification with regular audits and yearly reviews.

The Integra solution has been designed to comply with PCI requirements; in addition any integrating system using the 3cXml specification should also ensure the same compliance. In order to allow the integrator to meet these requirements the 3cXml protocol does not require the inclusion of track2 or track2-equivialnt (EMV ICC transactions) data after the initial authorisation response has been received from the acquiring bank, which is possible as Integra knows how to correctly treat and flag subsequent requests.

The PCI security can be guaranteed by the Integra solution. Card data is then read by PCI certified EMV terminals. Sensitive cardholder data is exclusively handled by the Integra secured solution and not handed over to the integrators. Instead Integra has the possibility to generate a deterministic card number substitution, also called token, and provide this one for later processing, e.g. in situations where the cardholder has left the location.

It is still possible to handle track2, PAN and other sensitive data between the integrator and Integra, but this is not suggested, as places the integrator more in scope of PCI, and might require a full PA-DSS certification.

7.3.3 Sensitive cardholder data configuration options and token

The Integra application can be configured to NOT send back any sensitive data in the message response. This shall be the configuration to look at first. The following configurations exist:

- Send back card number and track data as read from the card
- Send back NO card number and NO track data and NO expiry date
- Send back masked card number and NO track data

In addition Integra can be configured to send back a token string, in addition to the card number, or instead of the card number. The token is a hash created from either the card number string itself, or from some transaction data received at initial request from the integrator. The token is used as a reference for all subsequent requests in a payment cycle.

Another configuration possibility is to reject all requests containing card data in any card number request field.
For more detailed information on usage scenarios of the token contact our project engineering team.