HostPaymentData.hpp

struct HostPaymentData

- The Payment Data object is introduced to reflect the nexo/ISO20022 structure that encapsulates the data elements required to execute the required transaction, such as the Instrument used e.g. card and the authentication details (e.g. the actual signature or the encrypted PIN block if captured).

tring>

>

std::optional<std::string> cardBrand_, std::optional<std::string> cardproduct_, std::optional<OrderType> orderType_)

std::optional<std::string> billingStreet

std::optional<std::string> billingCity

ith defined boundaries, and a local government. This could be city or town.

std::optional<std::string> billingCountrySubdivision

dentifies a sub-division of a country which may be state, region or county.

std::optional<std::string> billingPostCode

- Customer Billing Address Postal Code Identifier consisting of a group of letters and/or numbers that is added to a postal address to

std::optional<std::string> billingCountry

alpha-3 country code.

std::optional<std::string> pinEncrytpedData

- Encrypted PIN. (Personal Identification Number). The PIN data associated with the Card holder or Customer (where known). This value should be an encrypted representation of the PIN. This is attribute ISO8583 DE52. This value can never be persisted or logged.

ne encrypted PIN content as a string.

std::optional<std::string> pinKeySerialNumber

- Authentication Online PinData EncryptedPINBlock The KSN is a constructed element that has multiple parts. In DUKPT 2009 this identification information is referred to as a Key Serial Number (KSN) sent by the initiator of the message to uniquely identify the derived key at the recipient. This is the first 5 bytes of the KSN. The structure should follow the standard:

```
Issuer Identification Number - 3 bytes
Merchant ID - 1 byte
Group ID - 1 byte
Device ID - 19 bits
Transaction Counter 21 bits.
```

std::optional<InstrumentType> instrumentType

rd, token or other... see InstrumentType

std::optional<std::string> maskedCardNumber

std::optional<std::unordered_map<std::string, std::string>> cardDataEmvTags

is a Verifone keys, Acquirer/Customer keys or none.

std::optional<std::string> cardDataEparms

- VCL AES-DUKPT encryption requires the use of eParms which is an additional blob of encrypted data associated with the encrypted

std::optional<CardEncryptionType> cardDataEncryptionType

std::optional<std::string> cardDataTrack2

blob.

std::optional<std::string> cardDataTrack2Ksn

onal. If encryption is based on the encryption used

std::optional<std::string> cardDataPan

std::optional<std::string> cardDataPanKsn

NOTE: This is optional. If encryption is based on the encryption used

std::optional<std::string> cardDataExpiry

std::optional<std::string> cardDataExpiryKsn

Entry NOTE: This is optional. If encryption is based on the encryption used

std::optional<std::string> cardDataCvv

std::optional<std::string> cardDataCvvKsn

Entry NOTE: This is optional. If encryption is based on the encryption used

std::optional<std::string> cardBrand

correspond to a consistent name, the list of constants is in CardInformation

std::optional<std::string> cardproduct

- The product name associated with the card brand. e.g. Classic, Standard, Gold, Platinum, World. This must correspond to an actual

std::optional<OrderType> orderType

- Order type see OrderType