BRITISH STANDARD

BS EN
15320:2007

Identification card systems — Surface transport applications — Interoperable Public Transport Applications — Framework

The European Standard EN 15320:2007 has the status of a British Standard

ICS 35.240.15

National foreword

This British Standard is the UK implementation of EN 15320:2007.

The UK participation in its preparation was entrusted to Technical Committee IST/17, Cards and personal identification.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 January 2008

ISBN 978 0 580 55709 5

Amendments/corrigenda issued since publication

Date	Comments

EUROPEAN STANDARD

NORME EUROPÉENNE

EUROPÄISCHE NORM

EN 15320

December 2007

ICS 35.240.15

English Version

Identification card systems - Surface transport applications - Interoperable Public Transport Applications - Framework

Systèmes de cartes d'identification - Applications pour le transport terrestre - Applications de transport public interopérables

Identifikationskartensysteme - Landgebundene Transportanwendungen - Interoperable Anwendungen für den öffentlichen Verkehr - Rahmenwerk

This European Standard was approved by CEN on 8 September 2007.

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

Management Centre: rue de Stassart, 36 B-1050 Brussels

Ref. No. EN 15320:2007: E

Contents

Page

List of Figures

List of Tables

Foreword

This document (EN 15320:2007) has been prepared by Technical Committee CEN/TC 224 "Personal identification, electronic signature and cards and their related systems and operations", the secretariat of which is held by AFNOR.

This European Standard shall be given the status of a national standard, either by publication of an identical text or by endorsement, at the latest by June 2008, and conflicting national standards shall be withdrawn at the latest by June 2008.

This document builds on the following standards to define an Interoperable Public Transport Application:

— EN 1545-1:2005, *Identification card systems — Surface transport applications — Part 1: Elementary data types, general code lists and general data elements*;

— EN 1545-2:2005*, Identification card systems — Surface transport applications — Part 2: Transport and travel payment related data elements and code lists*.

This document describes a foundation for a technology neutral environment for an Interoperable Public Transport Application within the confines of the definition of identification card systems. Nevertheless, interoperability cannot be maintained if different interface technologies are used by Machine Readable Cards within such a scheme. Consequently this document specifies the adherence to ISO/IEC 14443 Parts 1 to 3 as a necessity to ensure interoperability.

Amendments and enhancements to this European Standard will be made from time to time and published on the CEN website.

To the best of their knowledge the authors of this European Standard do not believe it infringes any commercial copyright, intellectual property rights or patents. However, CEN cannot guarantee this and shall not be responsible for any such infringements or claims, which will be dealt with according to CEN rules and regulations.

According to the CEN/CENELEC Internal Regulations, the national standards organizations of the following countries are bound to implement this European Standard: Austria, Belgium, Bulgaria, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.

Introduction

The Interoperable Public Transport Application defines the foundation and basic structure of a transport application primarily for ticketing for implementation on a Machine Readable Card that makes use of the Data Elements defined in EN 1545 and which may be made interoperable subject to commercial agreements between the parties involved and an exchange of specific implementation details. This has the effect that different operators will be able to read, interpret and handle Machine Readable Cards containing the application produced by others. Moreover, again subject to commercial agreements between the parties, it should be possible for a transport operator to write its ticket products to Machine Readable Cards issued by others that contain the application. Annex H discusses how legacy systems can interface with the application such that some level of interoperability may be achieved through a migration path to it.

This European Standard describes the basis of a public transport application resident on a Machine Readable Card as presented at the interface to a suitable terminal. In many cases where the card contains a processor, this interface will be between the card and the accepting device. In other cases, additional logic within the terminal application will be included in order to provide the necessary support. This is accomplished by mandating a logical abstract interface. The actual format of the data held on the card is not described by this European Standard. This format may be derived from a mapping of the data described in this European Standard to the card using an ASN.1 encoding rule.

This European Standard forms one part of a series relating to public transport which define the interoperable fare management system as shown in Figure 1.

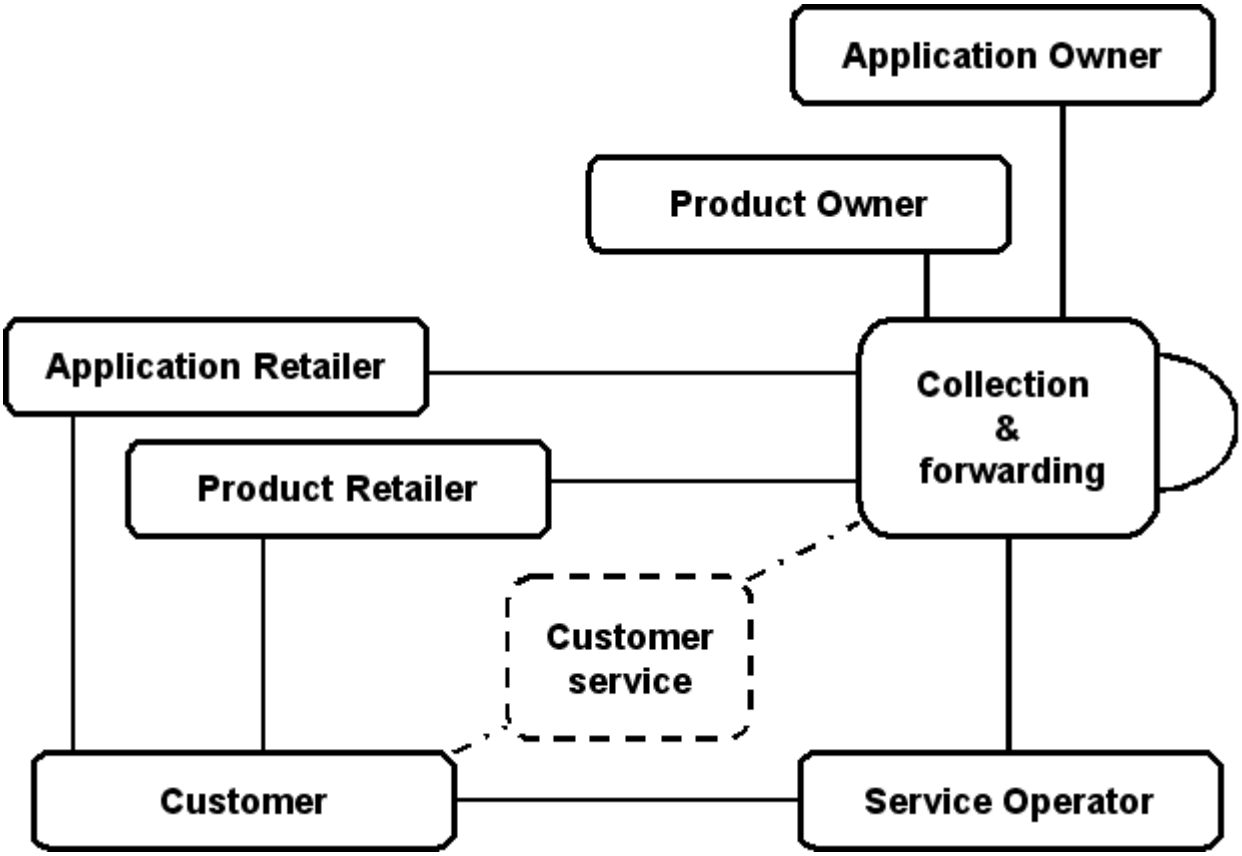

Figure 1 — Interoperable Fare Management system

This European Standard describes the basis of an environment which aims to achieve the following objectives:

— to provide a basis for offering machine readable interoperable tickets across the public transport network in Europe;

— to satisfy the demand for securing a seamless journey for the passenger allowing them travel with all participating operators, possibly in different networks and countries, using a single card while in the context of not inhibiting commercial competition.

This European Standard describes those components of the application necessary to support an interoperable environment including:

— accessing the Interoperable Public Transport Application;

— data structure and presentation;

— sizing and enumeration of data;

— data access methodology;

— security and access considerations;

— dealing with legacy systems.

1 Scope

This European Standard specifies sets of data presented at an interface, the card sub-system interface, in a structured form as well as the rules for dealing with that data to enable products such as tickets to be written to a Machine Readable Card in a manner which will minimise the amount of data to be held on the card while allowing an authorised party to be able to access and interpret the data easily and efficiently.

This is the basis for practical interoperability and as such, this European Standard forms the foundation of interoperability across systems subject to commercial agreements and interchange of details concerning how this European Standard has been physically interpreted. As part of this capability, the design of the data environment allows for the addition of new sets of data to represent new or modified transport products without compromising the ability of existing terminals to continue to handle all sets of data held on the card, whether or not they are to be interpreted and possibly used.

Associated with the data is the set of processes which applies to the data within the application. The inclusion of process in the standard means that similar data will be treated in a similar way by all external services and terminals leading to true interoperability that can be achieved and maintained through this European Standard. In addition, acknowledgement that the application specifies both data and process also implies that it needs to consider security both at the level of access rights to data and the security of the overall environment in which it operates.

The security related clauses in this European Standard define the minimum requirement of functionality necessary such that interoperability may be supported while protecting information stored within the application from unauthorised access and accidental or malicious damage. This European Standard defines an abstract card to card accepting device application interface which may be implemented, entirely at the card edge, or may include some logic in the card accepting device dependent upon the capability of the card. The view of security is similar in terms of an external system accessing, via the abstract interface, Machine Readable Cards, which may be just a card or a card – card accepting device combination. This means that security controls may exist in the card, the card accepting device or a combination of both. Additional descriptions of security architecture and expected implementation issues are described in Clauses 7 and 8.

This European Standard describes the minimum requirements for an interoperable transport application that may exist on a Machine Readable Card, either alone or together with other applications, and it is therefore a description of data sets and formats at the logical level. The abstract interface needs to support many Machine Readable Card varieties that conform to a contactless interface compatible with ISO/IEC 14443. ISO/IEC 14443 Parts 1 to 3 need to be supported. While this European Standard applies specifically to Machine Readable Cards, others may wish privately to use it with other customer media such as key fobs, subject to the customer media being able to interface with card acceptance devices supporting this European Standard where interoperability is required.

In terms of file structures, the data sets and data formats described in this European Standard are perfectly capable of being mapped onto a card conforming to ISO/IEC 7816-4. However, this European Standard does not define the card architecture, and the data formats and structures it defines are equally capable of being implemented on a pure memory card or a more complex multi-application card conforming to some other file format, subject to the card acceptance device supporting any required functionality that the card lacks in order to support the interface requirements of this European Standard.

This European Standard describes a generic logical model in ASN.1 format which may be mapped into a real environment using ASN.1 encoding rules such as BER and PER. However, it is recognised that certain overriding factors may affect the manner in which data is mapped onto real cards.

— Performance represented by transaction time is a critical issue in many transport applications. The PER encoding rules allow the physical data structure to be fixed and minimised in size using external Tag lists. For this reason it is expected that PER or some similar encoding rule will be used in practical implementations.

— Card data space limitations also mitigate towards the use of PER or similar encoding rules.

— Need to maintain compatibility, limited or full, with existing legacy systems and systems currently in development implies that specifically derived encoding rules may be specified to map the logical structures into the required format.

As a foundation for interoperability, this standard provides the basis for interoperability across instances of the application supplied by different parties subject to commercial agreement and exchange of details of the physical interpretations of the standard.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

EN 1545-1:2005, *Identification card systems — Surface transport applications — Part 1: Elementary data types, general code lists and general data elements*

EN 1545-2:2005, *Identification card systems — Surface transport applications — Part 2: Transport and travel payment related data elements and code lists*

ISO/IEC 7816-4:2005, *Identification cards -- Integrated circuit cards -- Part 4: Organization, security and commands for interchange*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
abstract syntax notation
form of notation used to describe data elements and processes, standard for CEN documentation

3.2
account
record of the current value and (truncated) transaction history of a product held on the 'back office' system of the 'product owner'

3.3
anonymous card
card which is not linked to a named holder, but which will still bear a traceable serial number

3.4
anti-tear
describes measures taken to ensure that any intentional alteration of data in the customer media during normal use does not lead to un-recoverable corruption of the customer media

3.5
application
instance of the Interoperable Public Transport Application resident on a Machine Readable Card or other customer media

3.6
application owner
entity which holds the application contract for the use of the application with the customer

3.7
application retailer
entity which sells and terminates applications, collects and refunds value to a customer as authorised by the application owner

3.8
Card Accepting Device
device which can interact with a card and exchange data with the card

3.9
card holder
person who owns the right to use the card

3.10
Charge to Account
facility/process for post-billing - rather than pre-payment or payment at the time of purchase (subtype of product)

3.11
check in – check out
holders actively validate cards when entering and leaving defined areas specified by a transport provider

3.12
concession
entitlement to a reduced (or zero cost) fare on the basis of age, condition or status

3.13
contract
expression of an agreement between two or more parties in the transport environment. It defines the conditions under which the user may use the services. Products such as tickets or entitlements represent a contract

3.14
customer media
entity which at least supports the same functionality as a Machine Readable Card but may be in a different form factor

3.15
data element
single store for an irreducible datum value (see EN 1545-1)

3.16
entitlement
entitlement or qualification for a service expressed as a product (a type of product template)

3.17
hot list
list of cards, applications, products or items of equipment where a transaction requires special attention

3.18
Interoperable Fare Management
encompasses all systems designed to manage the acquisition and use of fare products data in an interoperable public transport environment

3.19
interoperability
ability of systems to provide services to and accept services from other systems

3.20
journey
complete sequence of one or more journey legs (rides) required to achieve a specific purpose at a specific destination. This sequence may include the use of more than one vehicle and using more than one transport mode (see EN 1545-1)

3.21
key
binary string which is used to control access to an application or product, or which is used as the basis of encryption or during the calculation of Message Authentication Codes

3.22
loyalty
rewards for use of transport services

3.23
Message Authentication Code
computed field based on data in previous stated fields which allows a message to be verified as genuine

3.24
Machine Readable Card
token or entity that conforms to ISO/IEC 14443 Parts 1 to 3

3.25
profile
means to achieve interoperability between different card platforms

3.26
product
enables a customer to benefit from a transport service. It is an instance of a product template in the application. It is identified by a unique identifier. The Interoperable Public Transport Application distinguishes between four subtypes of a Product: CTA, STR, ticket and entitlement

3.27
product owner
entity which performs the functions of ownership (specifies pricing, usage rules and commercial rules), clearing and reporting

3.28
product retailer
entity which sells and terminates products, collects and refunds value to a customer as authorised by a product owner

3.29
product rules
product owner requirements (set of usage, pricing and commercial rules)

3.30
product specification
specification of function, data elements and security schema according to product rules

3.31
product template
technical master of the product specification for implementation. A unique identification (product template ID) is given to each product template by the registrar, triggered by the product owner

3.32
retailer
see: product retailer, application retailer

3.33
ride
component of a journey

3.34
route
reference to a single path through a transport network

3.35
seal
guarantee of authenticity

3.36
service operator
entity which provides the service to the customer against the use of a product

3.37
Stored Travel Rights
specialised form of closed e-purse for travel. (subtype of product template which uses transport tokens in place of currency)

3.38
ticket
entitlement for a journey (subtype of product)

4 Symbols and abbreviated terms

3DES Triple DES

AFI Application Family Identifier

AID Application Identifier

AM-DO Access Mode Data Object

ASN.1 Abstract Syntax Notation One

ATQ Answer to Request

ATS Answer To Selection

BER Basic Encoding Rules

CAD Card Accepting Device

CDS Control Data Structure

CEN European Committee for Standardisation

CICO Check in – Check out

CRC Cyclic Redundancy Check

CTA Charge to Account

DES Data Encryption Standard

FCP File Control

ID Identity

IEC International Electrotechnical Commission

IFM Interoperable Fare Management

ISO International Organisation for Standardisation

MAC Message Authentication Code

PER Packed Encoding Rules

REQA Request Type A

REQB Request Type B

RSA Rivat Shamir Adelman

SAK Select AcKnowledge

SC-DO Security Condition Data Object

SSS Security Sub-system

STR Stored Travel Rights

TLV Tag Length Value

5 Basic structure of application components[1)]

5.1 Data element

A data element in this European Standard is one of the data element entities described in the EN 1545. It is the lowest hierarchical entity of data discussed in this European Standard. Each element shall be associated with a tag and length descriptor to form a TLV data object according to ASN.1 conventions.

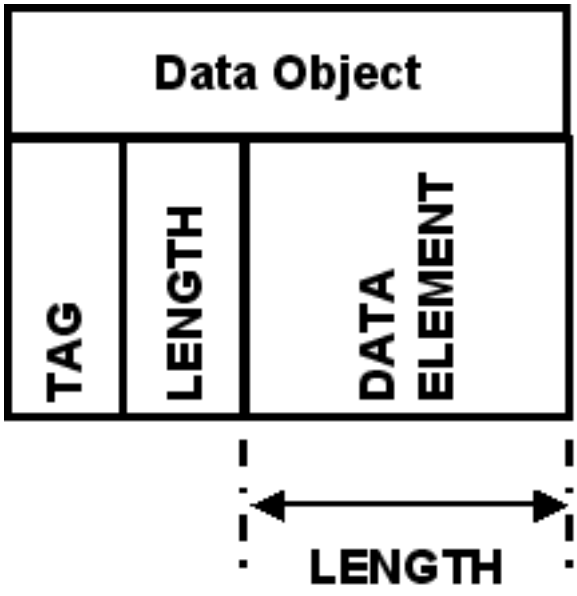

Figure 2 — Data element within a data object

Most of the data elements used in this European Standard are specified as ASN.1 primitive data elements and are originally defined in EN 1545 and enumerated in Annex C. In a few specific cases, the data elements used in this European Standard are formed as constructed data objects based on their definition in EN 1545.

1) The subclauses below describe the terminology used in this European Standard to refer to data.

5.2 Data structure

A set of data elements contained within data objects conveniently concatenated together as an ASN.1 constructed TLV object.

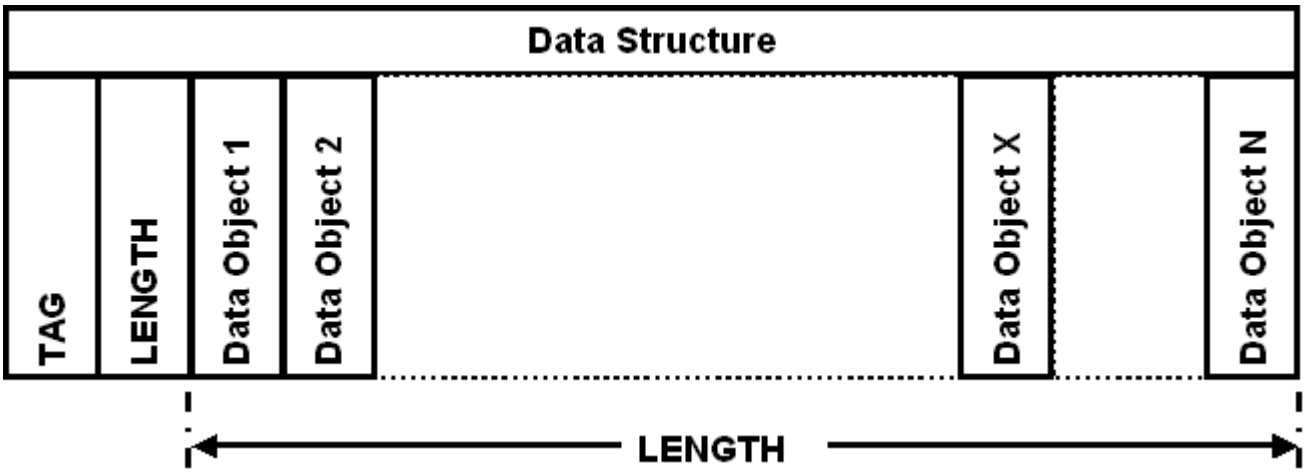

Figure 3 — Data structure

Data structures are classified for the purpose of convenience by type as follows:

M Mandatory (data structures that shall be present within a specific data group [see 5.3]);

A Additional (additional data structures that are normally created when the data group [see 5.3] or product is created. Data in these structures is normally not altered during the life of the data group which should be reflected in the security subsystem access rights applying to the data structure within the data group containing the data structure);

L Logging (additional data structures that are normally created during the lifetime of the data group [see 5.3] or product. Data in these structures may be fixed and not alterable once created or it may be updateable from time to time dependent upon requirement and the access rights applying to the access structure as controlled by the security subsystem).

5.3 Data group

A top level grouping of data structures concatenated together as ASN.1 a constructed TLV object to identify major groupings of data accessible as a coherent set through the Tag.

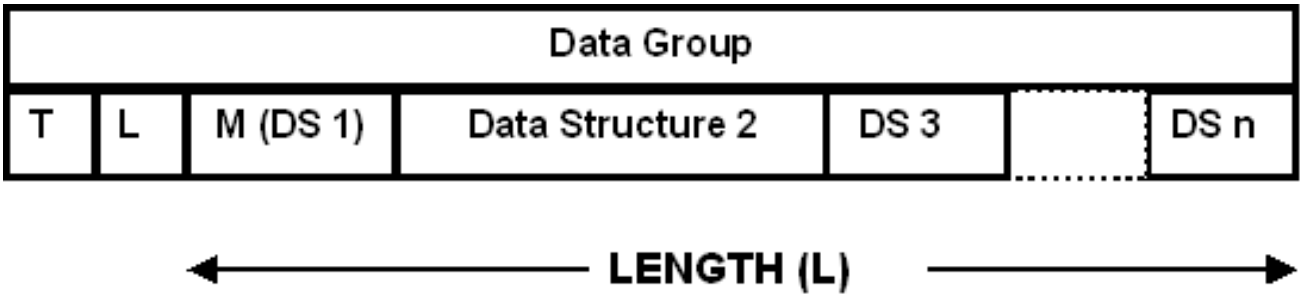

Figure 4 — Data group

The M data structure will always appear first in a data group, while A and L type data structures may appear in any order, however, A type structures will normally be created when the data group is created while L type structures will usually (but not always) be added after the data group has been created. It is usual therefore, but not mandatory, for the data group to be constructed in the sequence M, A…, L…

Once created and appended to a data group, data structures may not be deleted until the data group is removed from the Application in the card, for example when a ticket has been used.

6 Data groups

6.1 General

The construction of and access to data groups is defined here in a logical manner as presented at the interface and their realisation on a card may take many forms dependent upon the card architecture and encoding. There is no dependence upon or correlation to the definition of a card file structure as defined in ISO 7816-4 although the data group construction may be conveniently mapped into this architecture particularly to reflect the differing security and access requirements of the data structures comprising the data group.

The following data groups are defined in detail in Annex A:

— application environment;

— products;

— holder;

— event log;

— wrapper.

Data groups are linked as shown in Figure 5 and as further described in 6.2:

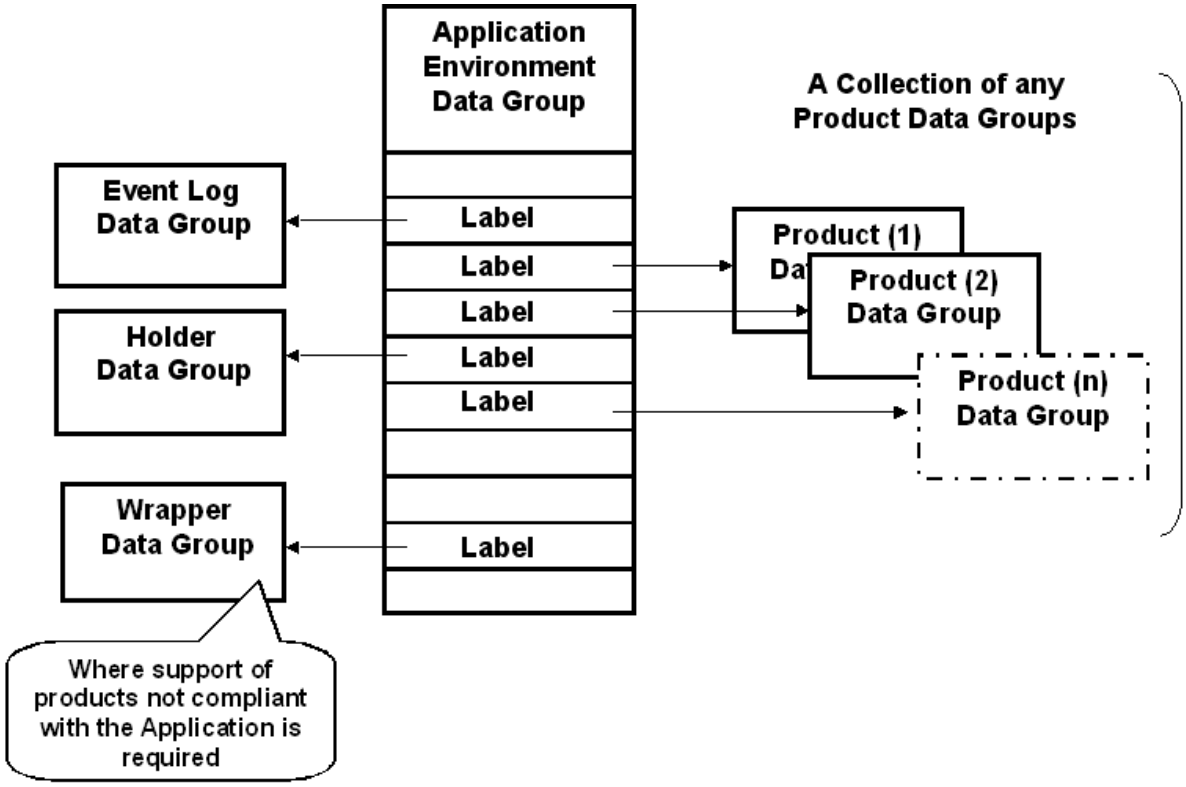

Figure 5 — Relationships between the data groups

6.2 Application environment data group

The application environment data group carries out two functions:

— provides the link to all other data groups within the application (that is, represents the application product directory);

— provides information pertaining to the application (specifically this instance of the application) on the card as interpreted at the interface.

6.3 Products

A collection of product data groups that identify products which may be created and removed during the life of the application. The product data groups are as follows.

— Stored Travel Rights;

— Charge to Account;

— customer entitlement;

— ticket.

Product data groups each consist of one or more data structures. The data structures consist of at least the relevant Type M data structure plus additional data structures of Type A and/or Type L created as required either when the data group was defined or at some later time.

Where a ticket type is such that the price is to be calculated at the end of a journey based upon the legs travelled, at each interchange an additional Type L structure may be appended to the active ticket data group. These Type L data structures may be added as necessary thereby increasing the length of the product data group, or its size may be limited by treating the added Type L data structures on a cyclic basis by replacing the oldest Type L data structures with new ones once the limit size has been reached.

Alternatively, where the ticket price is based solely on the number of legs travelled then at the start of the first leg a relevant Type L data structure will be included in the active ticket data group. At each interchange the count of legs field in this data structure will be updated. In this case, the additional Type L structure should have its access rights set to read/write.

6.4 Wrapper data group

The wrapper data group is intended specifically for migration of legacy systems. This data group will enable a legacy product such as a ticket to be included in an instance of the application and conform to this European Standard such that it may be created, recognised, accessed and deleted. However, processing of the legacy product once accessed is outside the scope of this European Standard and is left to the legacy system provider and the operators supporting it. This concept is further described in Annexes H and I.

6.5 Holder data group

The holder data group is optional where its absence will create an anonymous application, and its presence will define a personalised application. It should be noted however, that an application containing a holder data group may still operate anonymously simply by not accessing the holder data group.

The holder data group shall consist of one or more data structures comprised of at least the holder Type M data structure together with optional Type A and Type L data structures created as required either when the data group was defined or at some later time.

6.6 Event log data group

The event log is a cyclic set of data structures (records) of one type, the event data structure. The number of entries in the event log will always be at least two but may be more than two dependent upon the implementer's requirements and the envisaged Machine Readable Card space available. When the log is full the oldest entry is replaced by the newest. The implication of this is:

— event log shows a log of the most recent recorded "n" events;

— Event Log cannot be used to hold data that is to be retained across more than one event since one does not know when a record will be over-written. Thus one could not use this approach to hold a count of journeys but one might use it where the action taken at the current event depends upon the previous event;

— in order to make the event log most useful, it is important that it does not become "cluttered" with redundant records. For this reason, the event log should not record all events but only those events specifically designated as event log events;

— given the above point, if it is thought necessary to create a general event log that records all events, this will be a separately created and managed event log outside of the application. Most commonly, it would operate at the card level.

If required, and for differing purposes, events may be recorded both in the event log and in a data structure Type L associated with a specific data group, such as a ticket.

The method of implementation on the card depends upon the facilities available supporting this type of data group. In order to enable support in circumstances where the card does not support cyclic files and where such a construction has to be supported by the application, the event log data group contains a data element that shall be maintained by the application and is used to point to the most recent entry in the cyclic event log. Where the card does support cyclic files, this element may be ignored.

6.7 Linkage between data groups

There may be requirements for linkages to be formed between products. For example, a product may be linked to an entitlement, or a product's validity may be linked to the existence of another product within the application. This is achieved through the use of a dependency pointer data element which appears in a relevant data structure within a data group.

However, it shall be noted that there may be no method of nullifying this dependency pointer if and when the data group being referenced is removed. While the dependency pointer will never point to the wrong data group, it may point to a non-existent data group. For this reason, when processing data groups, any dependency pointers should be checked for validity before attempts are made to use them to reference other data groups. One method that may be used to resolve this is to include the dependency pointer in the logical view of the label so that as products are added and removed, any affects on dependency pointers can be checked and corrected before having to read whole product data groups.

The nature of the dependency pointer on a card is implementation dependent. However, at the logical level as described in this European Standard, it shall be mapped to the data group sequence number of the target data group. Interoperability between applications will be maintained through commercial relationships and an interchange of specific implementation approaches taken.

6.8 Data group life cycle

Data groups, as encoded, are created on a card and retained until they are no longer required and the space is required for other purposes, usually the creation of a new data group. Management of the data group life cycle will be a function of management of the application environment data group (directory) entries plus space management within the application workspace.

Data groups may also expand during their life cycle as new data structures are appended (Types A and L). How this is reflected in practice on a card is card dependent and outside the scope of this European Standard.

The life cycle/ retention of different data groups is as follows:

Application environment	- for the life of the application on the card
holder	- until explicitly released
product	
Stored Travel Rights	- until no longer required and explicitly released
Charge to Account	- until no longer required and explicitly released
customer entitlement	- until no longer required and the space is required
ticket	- until no longer required and the space is required
(wrapper	- until no longer required and the space is required)
event	- for the life of the Application on the card, space re-used on a cyclic record basis

6.9 Data group mandatory structures

6.9.1 General

Annex B details the mandatory data elements in the mandatory data structures (Type M) for each different type of data group. While there are some data elements within these that are unique to each data group type, as shown in Annex B, there are three sub-structures (composite data objects within the data group) that are common to all and referred to as the administrative data block. These support the requirements for uniquely identifying, security sealing and binding data groups to the application. They are:

— label;

— instance identifier;

— seal (optional).

6.9.2 The Label

6.9.2.1 General

A label is a data structure which uniquely identifies a data group. The label also includes the expiry date.

There are two types of label in every application namely:

— application label defined as the label of the application environment (a single instance per application);

— product or holder label

6.9.2.2 The application label

The application label identifies the application and is bound to the Machine Readable Card. It allows an application to be identified from among a multiplicity of applications. It provides information about the

application its owner and the application expiry date. The label holds sufficient information for a terminal application to determine whether this instance of the application may be accepted by the terminal application.

6.9.2.3 The product label

The product label is part of the mandatory data in a product data group. It is also included in the variable part of the application environment data group, where it allows a product to be identified among a multiplicity of products in any application environment. The product label holds sufficient information for a terminal application to determine whether any of the products present in this instance of this application may be accepted.

The label data may be duplicated in the product data group and the variable part of the application environment data group or it may be held in just one location and mapped into the other as shown in 6.9.5.2.

6.9.2.4 The holder label

The holder label is part of the mandatory data in a holder data group. Its placement and use is as described above for the product label in 6.9.2.3.

6.9.3 The instance identifier

Uniquely identifies the data group in order to identify a specific product and allow an audit trail that differentiates the many instances of a specific product type.

6.9.4 The seal

The use of a seal is not mandatory but where used, the following rules apply.

In order to detect unauthorised changes to data elements in the data group, a seal shall cover the group contents. The sealing concept covers all the data contained within a data group including the ASN.1 Tag and length fields.

The seal is presented at the interface and represents a calculation on the data either as written or as presented at the interface. The form of the seal and whether this seal is actually written to the card, or another seal is written based upon a different calculation, or the seal is omitted completely, is implementation dependent.

Groups of data structures that are fixed throughout the normal life of a data group may be sealed separately from any groups of structures that may vary during normal use. Since many data groups will in fact consist of multiple parts, some containing fixed data and some containing variable data, a separate seal may be created for each of the fixed and variable parts. This gives the product owner the option of preserving the integrity of the seal on one part of a data group whilst allowing other data to be modified and resealed by third parties, without the sharing of any security keys. The use of a single seal for some or all data groups is not precluded and may be the option selected in many implementations.

6.9.5 Data group layouts

6.9.5.1 Product/holder data groups

Figures 6 an 7 show the notional layout of a product or holder data group consisting of mandatory data structures and the optional data describing the product or holder, in structures as set down in Annex B.

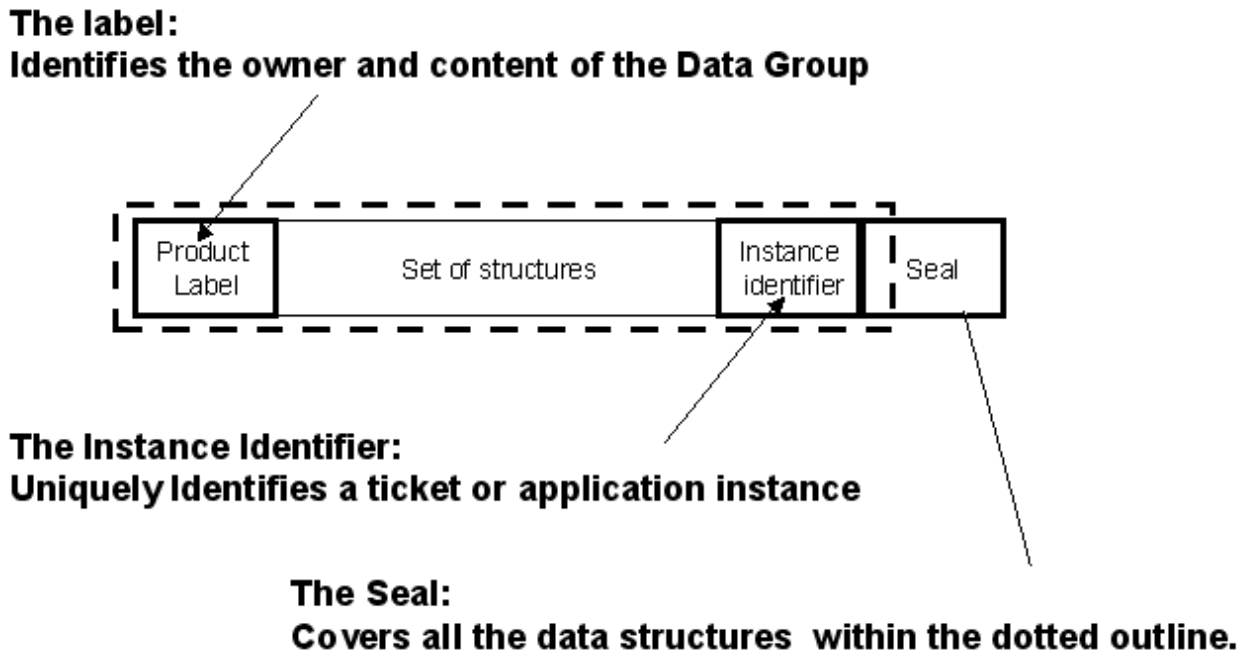

Figure 6 — Data group contents

Where the contents of a data group are logically split into fixed and variable data and separately managed, perhaps with different access rights applying to the fixed and variable parts, the data group may be shown as in Figure below. In this case the two instance identifiers may or may not be identical, while the two seals will be generated as a result of the data they are respectively sealing.

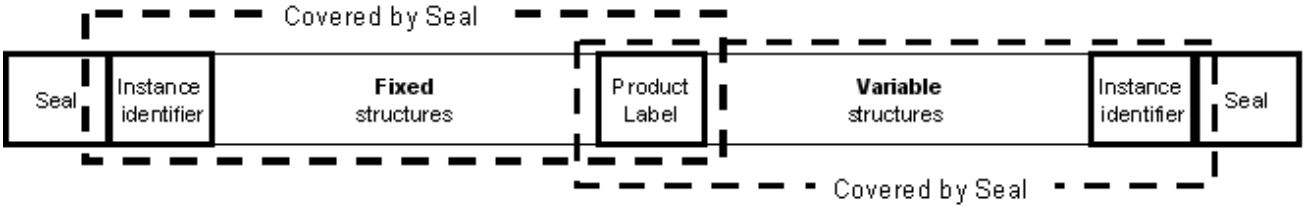

Figure 7 — Product data group with fixed and variable parts

6.9.5.2 Application environment data group

The application environment data group is constructed in the same manner as any other data group as shown in 6.9.5.1 above. However, the variable data in the data group is logically linked to the product and holder data groups identified by the label information present in each as shown in Figure 8:

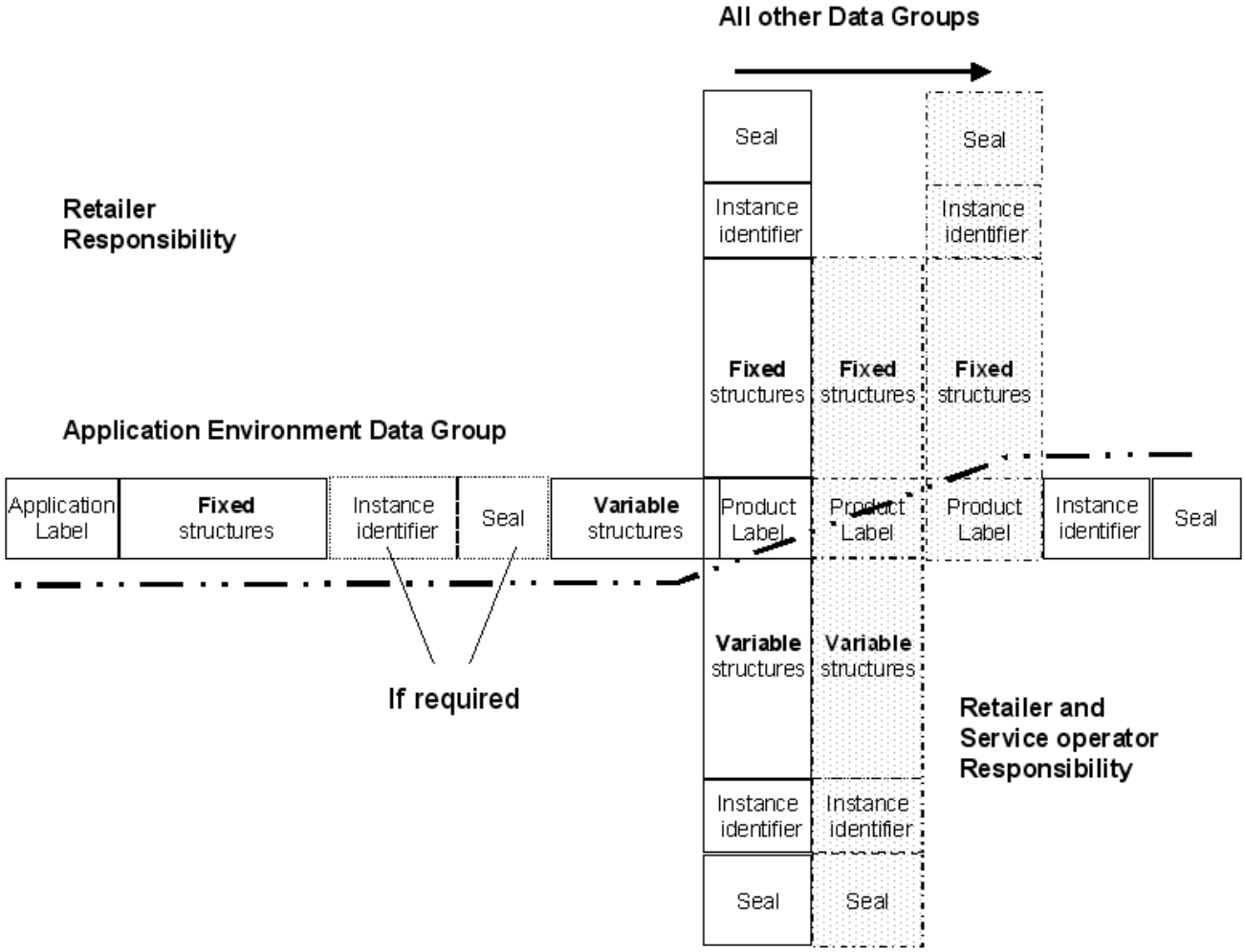

Figure 8 — The application environment links the data groups together

NOTE As shown, labels duplicate data defined within the mandatory (M) data structure for each data group. In terms of practical implementation only one copy of this data may be held in either the variable part of the application environment data group or the product data group and mapped to the other required location, or it may be duplicated.

6.10 Product data group priority

The priority data element is an optional data element in the instance identifier of a product data group. The use of this field, the values set in it, the rights to alteration and the ongoing management of it are entirely up to the implementer.

Access to the priority data element is via access to the product data group for each product existing at any time in an instance of the application. However, access may be speeded up if the instance identifier is mapped, as a logical view, into the application environment data group alongside the label. Effectively this moves the priority field into the 'product directory' of the instance of the application resident in the machine readable card.

7 The abstract interface

7.1 General

Data groups make up the products and the associated application label entries (directory). Data groups, as encoded, can be resident on a variety of Machine Readable Card platforms and other customer media that in turn interface to a variety of terminal applications.

This abstract interface provides standard services, which are not dependent on the capabilities of the card, in order that this European Standard shall provide the foundation for interoperability compatible with a large range of customer media. The distribution of roles inside the abstract interface between the Machine Readable Card and other customer media and the terminal application shall be adapted to the card's capabilities.

Two logical interfaces are defined in order to embrace the differing capabilities of the multiplicity of card platforms and products:

— logical interface 1: the card data interface;

— logical interface 2: the data group interface.

These interfaces define the connection between the terminal application and the Machine Readable Card and embrace the use of the security subsystem. Use of these logical interfaces between the teminal application and the machine readble card shall be mandatory for any application implementation.

The distributed nature of the relationship between the teminal application, Machine Readable Card and the security subsystem is illustrated in Figure 9.

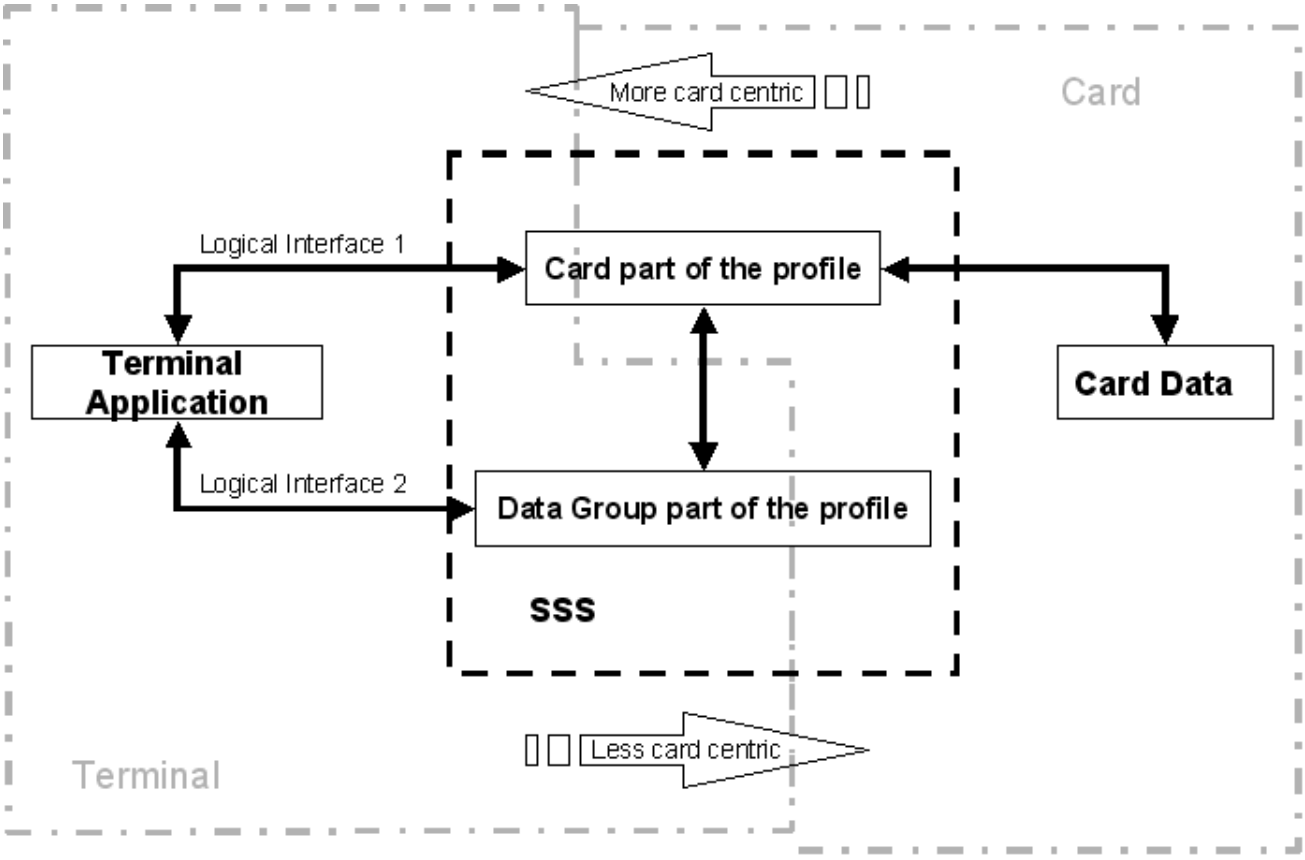

Figure 9 — Relationships between the logical interfaces, the SSS and the card and terminal

7.2 The Security SubSystem (SSS)

The Security SubSystem (SSS) embraces the secure parts of the terminal application and the secure parts of the Machine Readable Card application. It includes the functionality found in the card security management system and the Data Group security management system.

The SSS shall be implemented such that the card security management functionality shall be dynamically apportioned between the terminal and the card as appropriate for the capability of the Machine Readable Card.

Every manipulation of data involving the card and the terminal shall involve an SSS that is able to:

— act as a non volatile secure repository for application, product and transaction number generation;

— provide transaction records with secure messaging;

— check data groups for authenticity;

— provide secure access to data groups;

— hold all keys in secret;

— seal changed data group contents;

— provide secure storage and administration of profiles;

— implement secure messaging between the security sub system and the IFM manager for the acceptance and acknowledgement of updates to the security sub system contents.

The characteristics required of the security subsystem are for a secure application and memory that can hold cryptographic keys and scheme parameters. The secure application shall provide countermeasures to prevent:

— secret keys being revealed outside the application;

— security functions being modified;

— scheme parameters being changed without authority to do so;

— sequence counters from being reset.

The SSS should be accredited to formal security evaluation criteria.

NOTE It is assumed that all or part of these requirements may be met by using a secure device, for example in the form of a Machine Readable Card chip and application that can be easily fitted to every card accepting device.

7.3 Logical interface 1: the card data interface

7.3.1 General

This interface is described as a set of high-level function calls between the terminal application and the Machine Readable Card. The scope and detailed implementation of each function is conditioned by the security requirements and the profile of the card platform that is presented to the interface.

The function calls used by this interface consist of a scripted sequence of low level commands that are determined for a particular combination of terminal application and card platform. These differences are quantified in the appropriate Card Profile (CP(n)) that is stored in the card security management system part of the security subsystem. This interface is illustrated in Figure 10.

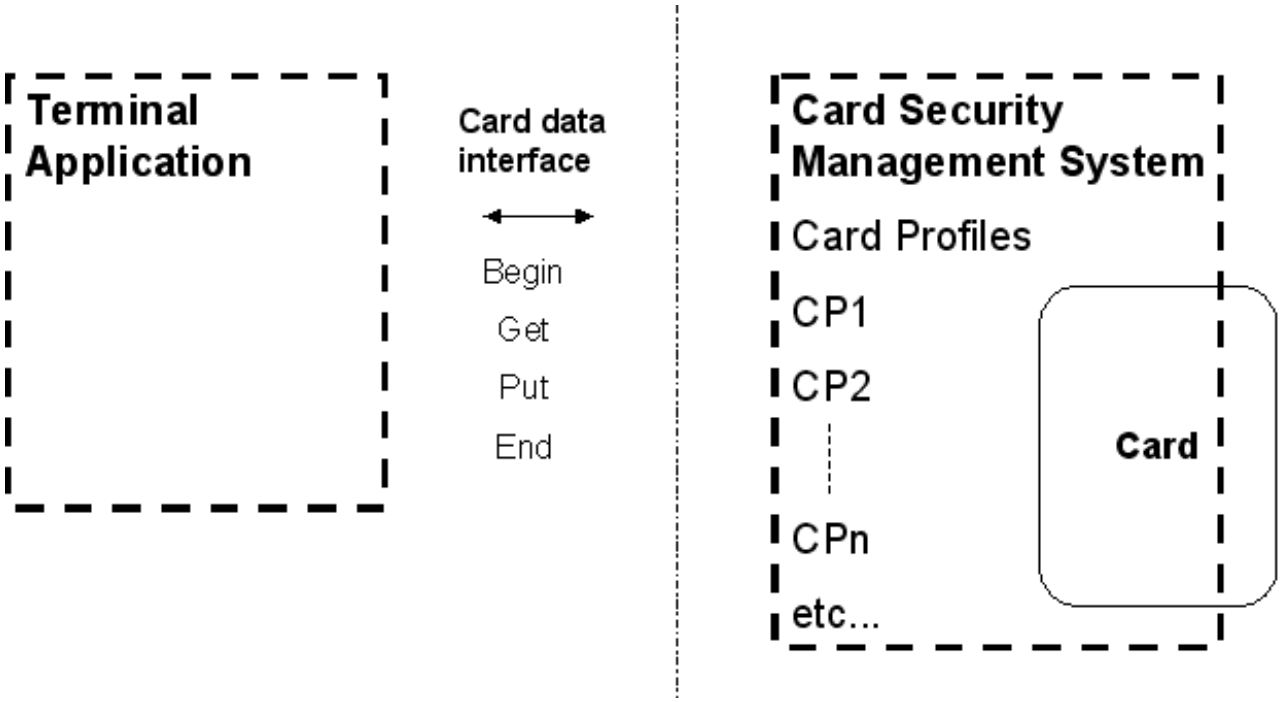

Figure 10 — Logical interface 1: the card data interface

7.3.2 Overview of the card data interface functions

Each function shall be implemented through a sequence of commands sent from the terminal to the card, dependant on the card technology, that when executed perform the high level functions described in Table 1.

Table 1 — Card data interface functions

ABSTRACT APPLICATION COMMAND[2]	DESCRIPTION
Begin	Select card profile, run profile, do security operations (authentication, diversification, RND generation, messaging architecture ….) (i.e. opens card session.)
Get	If permitted, authenticate and extract the message then transfer clear data to the terminal application.
Put	If permitted, generate the message to the card from the clear data.
End	End anti-tear management, reset all security states, deselect card profile, e.g. close card session.

2) The commands involve both the card platform and the security subsystem.

7.4 Logical interface 2: the data group interface

7.4.1 General

This interface is described as a set of high-level function calls between the terminal application and the data group being processed. The scope and detailed implementation of each function is conditioned by the security requirements and the profile of the data group being processed.

The function calls used by this interface consist of a scripted sequence of low level commands that are determined for a particular combination of terminal application and data group. These differences are quantified in the appropriate Data Group Profile (DGP(n)) that is stored in the data group security management system part of the security subsystem. This interface is illustrated in Figure 11.

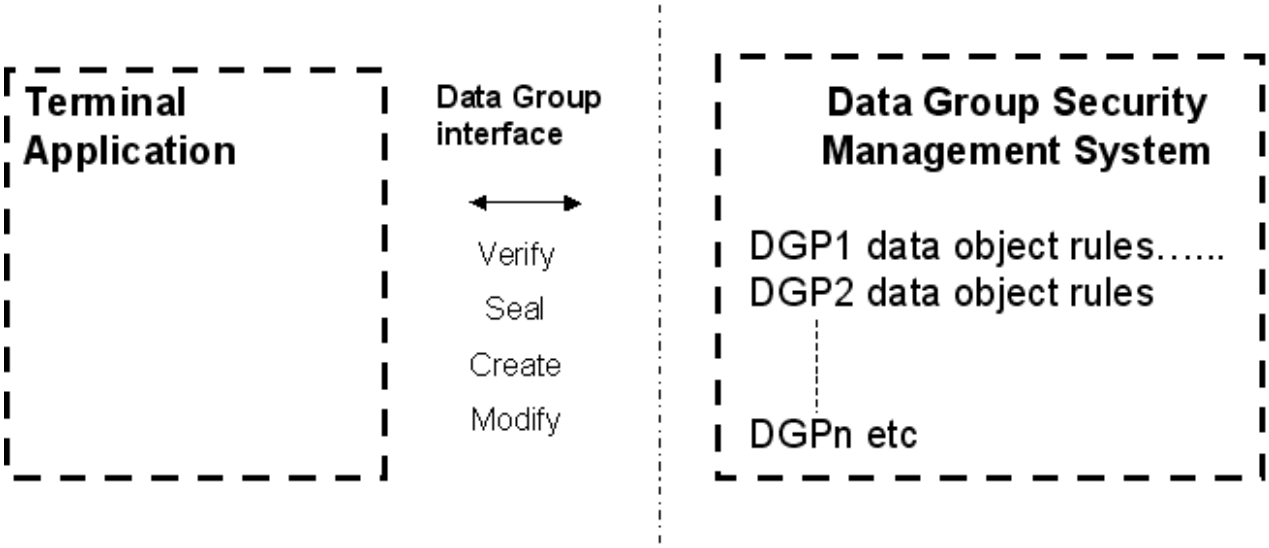

Figure 11 — Logical interface 2: the data group interface

7.4.2 Overview of the data group interface functions

Each function shall be implemented through a sequence of commands sent from the terminal to the security subsystem and / or the card, dependant on the card technology, that when executed perform the high level functions described in Table 2.

Table 2 — Data group interface functions

ABSTRACT APPLICATION COMMAND	DESCRIPTION
Verify	If permitted, select keys, verify seal, (decrypt dataset) apply data object rules …. and report.
Seal	If permitted, (encrypt dataset) select keys, apply data object rules, generate seal…and return seal.
Create	If permitted, create data group add unique numbering, apply data object rules.
Modify	If permitted, modify contents, add unique numbering, apply data object rules.

7.5 Application related commands

7.5.1 General

The application shall accommodate many different card types. The main difference between these types is the capability of the technology and the characteristics of the access to the application. Examples of card types range from small memory cards to highly sophisticated micro controller cards.

For micro controller cards, many general standards and specifications use ISO/IEC 7816 as their base and offer an extended set of interoperable commands for reading, updating and administrating data elements and data structures. Methods for application selection and security related functionality are also provided by those general standards and specifications.

At the other extreme, a small memory card generally offers only low level read and write commands. An application related interpretation of the memory content has to be carried out outside of the card. Security related functionality cannot normally be provided by the card. In this case most of the security application has to be performed in the terminal or in the back office.

This European Standard does not describe the detail of every command to be used with Machine Readable Cards but indicates the higher level abstract commands.

The implementation of abstract commands pertaining to the card data interface shall, wherever possible, make use of sequences of commands that are covered by exisiting ISO/IEC standards.

7.5.2 Abstract commands and use cases

Table 3 identifies application related use cases that shall be available on an abstract level of the Machine Readable Card. Some of these use cases may be constructed from several card commands; some of them may be performed with one single command or may also be an application specific card command. The listed use cases are not necessarily applicable to all types of Machine Readable Card containing an instance of the application. There may be cards, which fulfil only a subset of these use cases.

Table 3 — Application activities and use cases

Activities	Use-Cases Description	Logical Interfaces 1/2	Abstract Application Commands	Example Standardised Commands for µController	Example Proprietary Commands for simple memory cards
Administration	General activity to change the content of the card:	1	GET PUT VERIFY SEAL CREATE MODIFY	Create Application Create/Delete EF/DF Append/Update Record Read/Update Binary Get Data/Put Data Activate/Deactivate/Terminate	Write Sector Read Sector Authenticate (part) Present password None
Authorisation	Grant right for further activities:	1,2	VERIFY	Verify Card Holder External/Mutual Authentication Verify Signatures Secure Messaging (Trusted Channel), Derived Session Keys	Read Sector Authenticate (part) Present password None
Activation	Command to activate an instance of the application or data group	1	PUT	Activate Application Activate File	Write Sector Authenticate (part) Present password None
Validation	Check or cancel a ticket, entitlement	1	GET PUT	Read/Update Record Read/Update Binary Get Data/Put Data	Write Sector Read Sector Authenticate (part) Present password None

Table 3 (*continued*)

Activities	Use-Cases Description	Logical Interfaces ½	Abstract Application Commands	Example Standardised Commands for µController	Example Proprietary Commands for simple memory cards
Creation	Create a new instance of the application, data groups, data structures, data elements	1,2	CREATE SEAL	Create Application Create EF/DF Append/Update Record Write Binary	Write Sector Authenticate (part) Present password None
Deletion	Deletion of an instance of the application, data groups, data structures or data elements	1,2	MODIFY PUT	Delete Application Delete File Update record Write Binary	Write Sector Authenticate (part) Present password None
Renewing	A card or product where the antitlement is terminated will be renewed	1,2	MODIFY PUT	Update Record Update Binary Create Application Create EF/DF Append/Update Record Write Binary	Write Sector Authenticate (part) Present password None
Replacing	A lost card will be replaced	1,2	CREATE PUT	Update Record Update Binary Create Application Create EF/DF Append/Update Record Write Binary	Write Sector Authenticate (part) Present password None
Execute Transaction	General activity with the card data, which are not covered by the other use cases	1	PUT GET	Read/Update Record Read/Update Binary Get Data/Put Data	Write Sector Read Sector Authenticate (part) Present password None
Cancelling	Cancellation of a ticket	1	MODIFY PUT	Read/Update Record Read/Update Binary Get Data/Put Data	Write Sector Authenticate (part) Present password None

Table 3 (*continued*)

Activities	Use-Cases Description	Logical Interfaces 1/2	Abstract Application Commands	Example Standardised Commands for µController	Example Proprietary Commands for simple memory cards
Reinstate	Replace a data group	1,2	CREATE MODIFY PUT	Update Record Update Binary Put Data	Write Sector Authenticate (part) Present password None
Modification	Activity to change the content of an existing data element	2	MODIFY PUT	Update Record Update Binary Put Data	Write Sector Authenticate (part) Present password None
Ticketing	Activate a ticket at commencement of a journey	1	MODIFY PUT	Read/Update Record Read/Update Binary Get Data/Put Data	Read Sector Write Sector Authenticate (part) Present password None
Termination	Activity to stop the usage of an instance of the application or data group, without deletion	1	MODIFY PUT	Update Record Update Binary Deactivate Put Data	Write Sector Authenticate (part) Present password None
Inspection	Activity to add or change data elements on inspection	1	GET PUT	Read/Update Record Read/Update Binary Get Data/Put Data	Read Sector Write Sector Authenticate (part) Present password None
Get Product	Read out the entitlement, read out the ticket	1	GET	Read Record Read Binary Get Data	Read Sector Authenticate (part) Present password None
Get Receipt	Get an acknowledgement of a transaction	1	GET	Read Record Read Binary Get Data	Read Sector Authenticate (part) Present password None

7.5.3 A representative transaction

A representative card transaction shall involve the start and finish of a card session and the implementation of one or more use cases in between. The terminal application shall perform the application abstract commands using scripts of low level commands that are card platform and use case dependent.

Figure 12 below illustrates such a session for representative microprocessor and memory cards. Sessions start and finish with the BEGIN and END commands respectively.

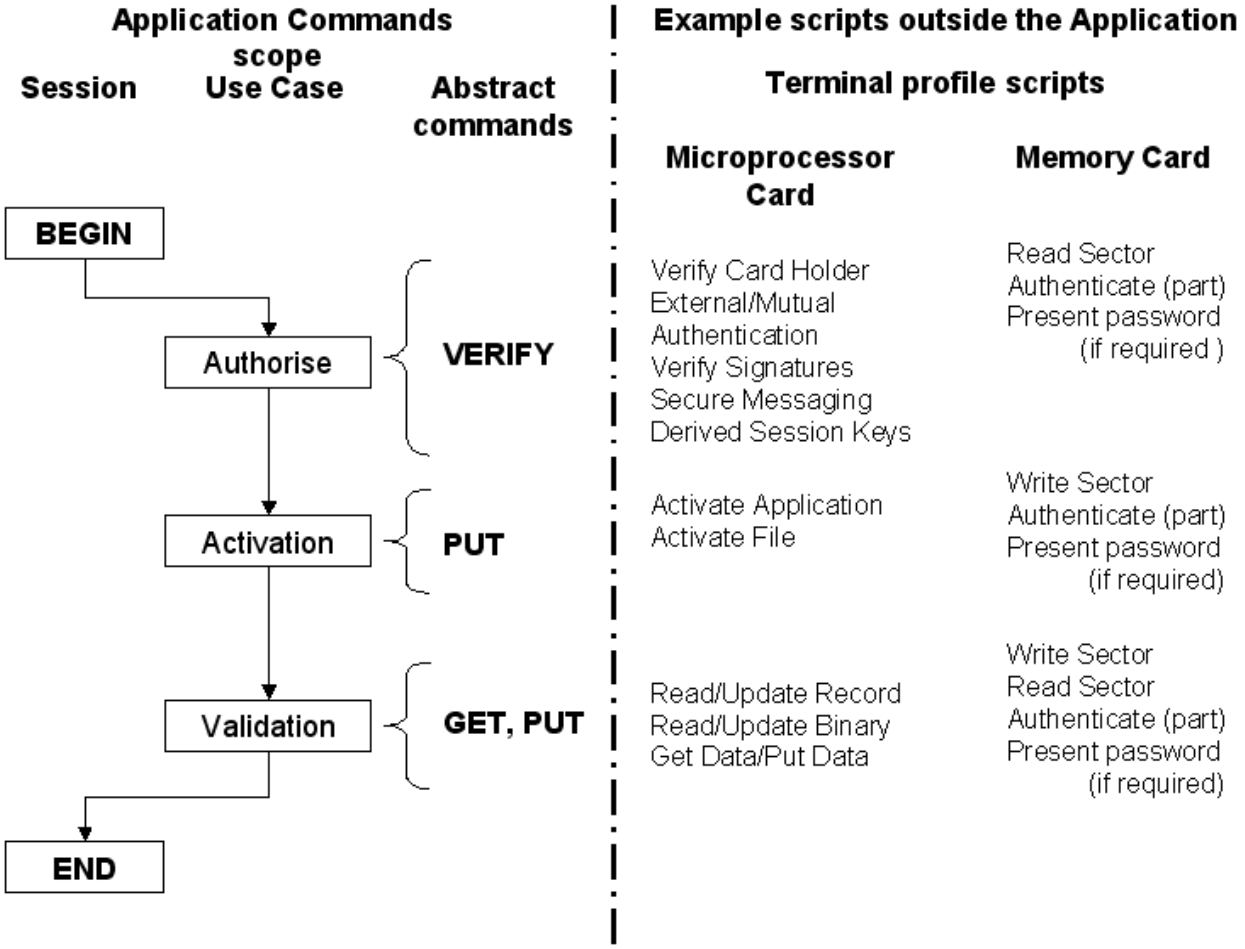

Figure 12 — Representative application command flow

7.5.4 Relationship with the logical interfaces

The state diagram in Figure 13 shows the terminal application states 3, 4 and 5 that are covered by the abstract command interface in the normative part of this European Standard. States 1 and 2 are detailed in the informative Annex G.

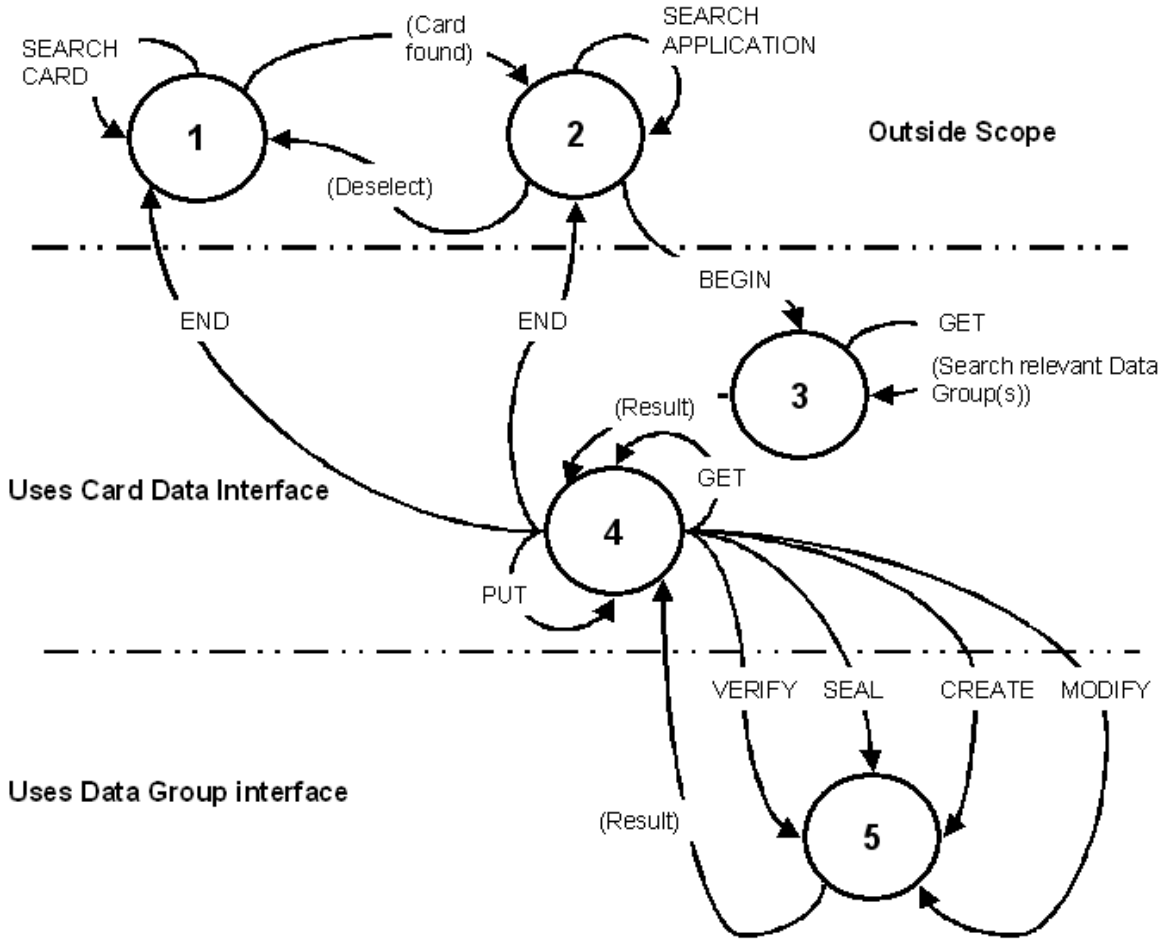

Figure 13 — Application states

8 Application security

8.1 Introduction to the security architecture

This clause covers the card to card accepting device interface and describes a security architecture that allows application products to be placed on cards ranging from memory only to high-end microprocessor. In conjunction with the security subsystem it forms part of an end-to-end overall security architecture for the application.

The agreements between the different business partners influence the security features required of a Machine Readable Card containing the application. It is to be noted that the security features dictate the requirements of the terminal, the security subsystem and possibly the background system. Consequently, it is impossible to identify just one security system for an instance of the application since different schemes could use different card technologies with different implementations. This clause and its subclauses describe security features, which may be used with different Machine Readable Cards.

The architecture set down herein separates the logical security requirements for application products from the Machine Readable Card and allows for access rights to data structures within the data groups to be

determined mutually between the terminal application and the card. The role of the terminal[3)] and the capability of the card are used to determine the split of security subsystem functionality between the terminal and Machine Readable Card by means of a profile negotiation.

Logical interfaces 1 and 2, as described in Clause 7 reflect this split of functionality such that access to the data groups is controlled by the card platform attributes (passkeys, mutual authentication…et al).

8.2 Requirements and Approachs to the Security Architecture

8.2.1 General

The application security architecture shall provide:

— confidentiality;

— integrity;

— authenticity;

— non-repudiation;

— anti-cloning;

— auditability.

The following sub-classes detail each requirement in turn from which a selection can be made for an SSS / Machine Readable Card combination.

NOTE For microprocessor cards many security features are described in detail in ISO/IEC 7816-4 and ISO/IEC 7816-9.

8.2.2 Confidentiality

Confidentiality is important for a system to guarantee that only an authorised entity is able to access confidential information. This can be provided by different means, many are well defined in existing Machine Readable Card standards documents and for reference, some of the methods are outlined below:

— restriction of access to data elements to only those parties with permission to do so, this may be implemented by using password, mutual authentication processes or other methods;

— encryption of sensitive data elements in order to maintain commercial secrecy or in the case of personal data, privacy;

— classifying data element(s) as “read only” to the cards internal application. In this case this data may be conditional to the successful completion of a function.

8.2.3 Integrity

Integrity can be achieved using many techniques; some of the methods are outlined below:

— use of a Luhn check digit is a basic method applied to some ISO numbering schemes to detect miss transcription of numbers;

3) Validator, ticket office machine, balance checker….etc.

— use of a checksum (ISO/IEC 14443). This technique is used to detect data corruption errors. It does not however give any guarantee of authenticity.

The contactless interface (as described in ISO/IEC 14443) is, by its very nature, prone to interruptions when in use; either from adjacent intermittent interference or, more commonly, by the removal of a Machine Readable Card from the CAD during a transaction. Anti tear methodology shall be used to alleviate this problem. In this case either the hardware of the Machine Readable Card or a software equivalent in either or both of the Machine Readable Card and the terminal ensures that valid card data remains even if the card loses power during the modification of data elements.

8.2.4 Authenticity

The provision of methods whereby the authenticity of data elements can be determined and may include the following techniques:

— sealing a data group provides a guarantee of authenticity that the contents of the data group have not been tampered with. The level of confidence in the seal will depend on the algorithms and keys used in its implementation;

— Applying Message Authentication Codes (MACs) to data transferred. This adds confidence that messages have not been altered whether by accident or on purpose.

The card and terminal may authenticate each other. This may take many forms for example:

a) microprocessor cards

    1) internal, external or mutual authentication algorithms, zero knowledge proofs;

b) memory cards

    1) may apply proprietary algorithms to authenticate an entire card or on a sector by sector basis.

8.2.5 Non-repudiation

In any open system the ability to confirm that transactions have been correctly reported and not removed or substituted is essential. Techniques to give a level of non-repudiation of data, used in conjunction with the seal, may include:

— uniquely numbering product and application instances throughout the application to provide the ability to detect duplicate or missing instances of data groups;

— using sequence counters in products where the data content varies with time to enable the ability to track for correct operation, for example the number of rides being incremented out of sequence on a multi ride product.

8.2.6 Anticloning

One of the simplest card frauds is to copy an entire card onto another empty card (cloning). Measures to deal with cloning may include:

— diversification of card access and / or authentication keys. In this case no two cards share the same access keys thus the straight copying of data from one card to another is prevented. Diversification should, where possible, use unique data that is permanently etched into the cards memory, or data that is known to be unique and written into one time programmable memory on the card;

— diversification of data group seals. By ensuring that any data group seal is diversified by data from the physical card platform the seal effectively becomes locked to that platform. Thus if that data group is copied to another card platform, seal verification will fail;

— supply of “application ready” but un-personalised cards should be restricted to only those that need them. All Machine Readable Card failures should also be accounted for.

Message cloning, namely the replaying of messages between card and terminal for disruptive or fraudulent purposes, can be prevented by the use of techniques known as secure messaging. In this case a secured session is set up between the card and the terminal such that communication in either direction carries a message authenticator unique to that session. Secure messaging may be implemented in many different ways but commonly a MAC is applied to every message using a key derived from the mutual authentication process. Other methods involve the encryption / decryption of data sent across the communications interface.

8.2.7 Auditability

Auditability within an application environment may be achieved by applying the previously specified requirements and maintaining, where appropriate, logs of transactions and events that occur in an application environment. This will make use of the unique numbering of:

— applications;

— products;

— transactions;

— lists.

This requires both a registration authority and subsequent governance, both of which are primarily the responsibility of the IFM and shall be included in the relevant parts of the IFM documentation as defined elsewhere. However the numbering of products created by an offline retail terminal shall be managed securely and entrusted to part of the SSS functionality.

8.3 Constraints in supporting a range of card platforms

The security architecture shall be flexible enough to support the addition of products on a variety of Machine Readable Cards. Some typical examples are given here:

a) small memory card / contactless ticket:

   1) no on-board security protection for access control;

   2) does not support ISO/IEC 14443–4;

   3) does support an embedded serial number;

b) secured memory card

   1) access is controlled by proprietary security methodology using short keys;

   2) does not support ISO/IEC 14443–4;

   3) does support an embedded serial number;

c) microprocessor cards:

   1) access control using mutual authentication;

2) may support ISO/IEC 14443-4;

3) usually supports published security algorithms (i.e. DES, 3DES, RSA….);

4) may support secure messaging;

5) may support access control at the data element level.

The card profiles installed in the card security management system allow the terminal application to adapt its low-level card drivers to match the security requirements of the platform being used. To resist tampering some of the functions used shall be embedded in the SSS.

8.4 Requirement to support a range of data group sealing techniques

All data groups may include a seal, which if generated by accepted cryptographic methods provides a guarantee of authenticity to the dataset from which it is generated. This guarantee covers the data group independently from, but may be linked to, the card platform upon which the data group resides. Seals cover all the data objects in the data group including data objects containing the instance identifier and label data elements. Even when structures transported by the card media are sealed, the control of the validity of seals is as determined by the security scheme accepted on the network and may not be mandatory in all equipment. Seals may take a variety of forms such as:

a) no seal present:

1) neither data integrity nor authenticity are proven;

b) CRC:

1) data integrity is verifiable;

2) authenticity cannot be verified;

c) MAC or digital signature:

1) data integrity is verifiable;

2) authenticity can be verified;

d) MAC or digital signature diversified by the identity of the customer media

1) data integrity is verifiable;

2) authenticity can be verified;

3) cloning is effectively resisted.

8.5 Unique numbering

Unique numbering shall be used to facilitate “end to end” security; management of action and hot lists and provide audit trails of all application activity. Every instance of every product and the transactions generated shall be uniquely identified in an instance of the application. The method of doing this is described in detail in the relevant IFM document.

To fulfil this requirement a data structure known as the instance identifier shall be included in every product and every application. This structure is coded such as to uniquely identify an individual instance of a product or application amongst a multiplicity of application environments and is fundamental to the provision of an irrefutable audit trail of product and application retailing and usage.

Because of its crucial role in the implementation and governance of any application environment the instance identifier shall be included as part of the product and application environment and be covered by the seal.

The application may avoid unnecessary duplication of product, serial and iteration numbers, elsewhere in the data group, by using the instance identifier data structure.

NOTE The responsibility for specifying numbering of Owners, transaction data and the unique identification of the security sub-system is with the specification of a full interoperable fare management system (IFM), which defines the end-to-end operation including the Machine Readable Card, and of which the application forms a part. Other relevant standards cover the specification of numbering.

8.6 Security related card information

8.6.1 General

The application consists of a number of data groups, each consisting of different data structures and data objects containing application specific data elements. The access to each data structure/ data element shall be protected. The necessary access conditions for a single data structure/ data element are derived from the business rules agreed between the different partners.

In the simplest case the access conditions are implicitly known by a terminal and the Machine Readable Card and no additional information stored on the card is necessary.

In general, some control information may be held on the Machine Readable Card, for one or more of the following reasons:

- card is able to control the access and the functionality required;
- business rules may change within the life time of a card;
- card and terminal need a flexible method of working with cards of different schemes;
- terminal profiles have to be extended in the field;
- security system uses an evolving key management and changing key versions.

The following subclauses indicate a flexible means to let card or terminal have access to control information which may be optionally implemented.

8.6.2 Control Data Structure

The Control Data Structure (CDS) is an optional logical data structure that may be present in any data group and which contains security and management information pertaining to any data structure or data elements present in this data group. If a CDS is absent from a data group, the access conditions are implicitly known to the terminals. An existing CDS in the card should overrule any terminal profile information.

A CDS is usually an A-type data structure, established when its data group is created, and not altered during the lifetime of the data group.

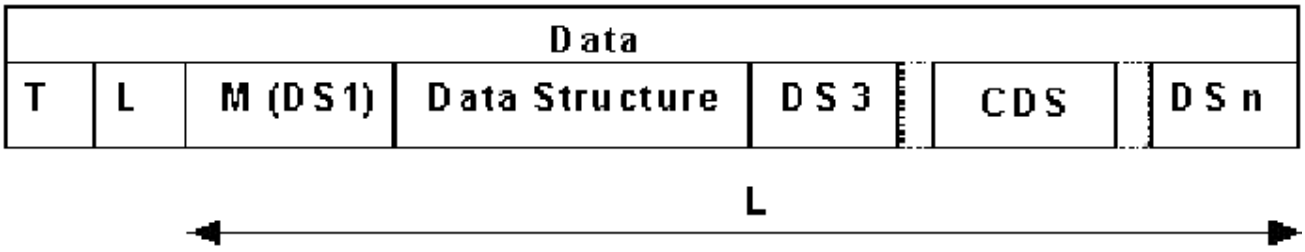

Figure 14 — Data group Control Data Structure

8.6.3 Format of the Control Data Structure

The format of a CDS is derived from the definitions of the access conditions in ISO/IEC 7816-4. A binding access condition for a data structure comprises a data object for describing the applied functionality (ISO/IEC 7816-4 Access Mode-DO or Access Mode Byte) and a data object describing the security condition (ISO/IEC 7816-4 Security Condition-DO or Security Condition Byte). The compact or expanded format might be applicable.

An entry of CDS contains a data element consisting of:

— reference to a specific structure/element in the data group;

— access condition.

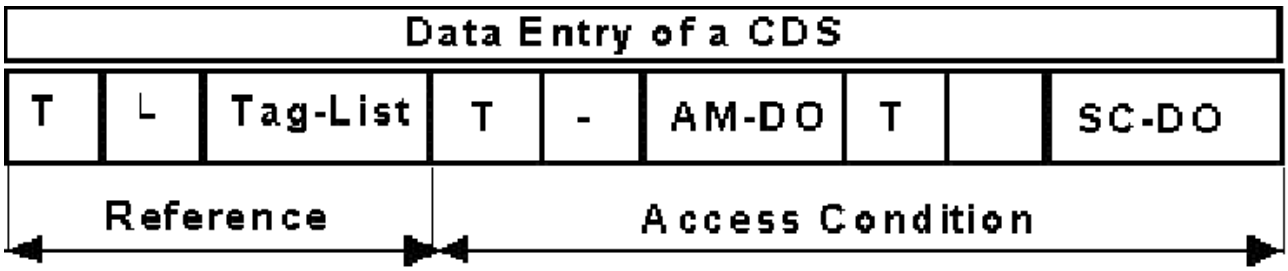

Figure 15 — A Control Data Structure entry

This description is also useful to protect an entry in the CDS.

ISO/IEC 7816-4 defines the general coding of an AM-DO and a SC-DO.

8.6.4 Extension of access mode coding

The definition of an AM-DO can also be used to define the allowed functionality to a specific data structure/element which may be in the form of a standard command or a proprietary command related to the instance of the application. For example, a data element used as a counter should only be manipulated by an INCREASE/DECREASE command. Other commands are forbidden and not applicable to this data element.

The definition of the access mode byte in ISO/IEC 7816-4 can be adapted to the needs of the application as shown in Table 4:

Table 4 — Access mode byte specification

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	-	-	-	-	-	-	-	Commands according definition of ISO/IEC 7816-4:2005, Tables 16 to 19
1	-	-	-	-	-	-	-	b7-b4 Application specific definitions
-	1	-	-	-	-	-	-	Transport Application Specific Command 1
-	-	1	-	-	-	-	-	Transport Application Specific Command 2
-	-	-	1	-	-	-	-	Transport Application Specific Command 3
-	-	-	-	1	-	-	-	Transport Application Specific Command 4
-	-	-	-	-	1	-	-	Command according definition of ISO/IEC 7816-4:2005, Tables 16 to 19
-	-	-	-	-	-	1	-	Command according definition of ISO/IEC 7816-4:2005, Tables 16 to 19
-	-	-	-	-	-	-	1	Command according definition of ISO/IEC 7816-4:2005, Tables 16 to 19

The usage of an AM-DO with APDU header definitions enables the process for specifying further application specific commands.

9 Profiles

9.1 General

Profiling is a means whereby interoperability can be achieved between different card platforms. This clause describes the different approaches and formats used in the application to establish the profile and the contents thereof.

Dependent upon the Machine Readable Card capabilities, the terminal and its security subsystem have to fulfil different tasks. Different card platforms require different methods by which the terminals can establish the correct profile to use. Interorability between different instances of the application may be achieved through commercial agreements and exchange of implementation details.

9.2 Request to a profile

A profile shall inform the terminal about:

a) application structure:

   1) how to retrieve the directory of contents;

   2) how to access to a specific product;

b) Machine Readable Card platform:

   1) memory card (simple);

   2) manufacturer specific memory cards with additional features;

3) µ-controller card;

c) command structure:

1) list of available command sequences;

2) use cases built by macros;

d) security information:

1) relation between data groups/data structures/data elements and access rights;

2) key version or keys in use.

9.3 Distinguishing a profile

9.3.1 General

Terminals identify profiles with a unique profile identification number. In general a profile ID should be unique for a specific card type, product and functionality. To restrict the number of profiles, a profile as implemented could refer only to a specific card and may contain all the required information and scripts for all different products and functions related to this card.

9.3.2 Registered profile IDs

Profile IDs shall be centrally registered such that any terminal supporting the application can unambiguously identify the necessary profile to use.

9.3.3 Definition of the profile ID

The profile ID is data element consisting of several bytes interpreted by the terminal. For different cards it could be located in different position in the card, e.g. in the historical bytes of the ATS[4)], in the FCP[5)] of the application dedicated file, at well defined sectors in a memory card or in a data group in the application itself. The basic structure is shown in Figure 16.

TAG	LENGTH	CONTENT
TProfile	LProfile	**Profile-reference**

Figure 16 — Profile ID structure

Certain card platforms may not provide the Tag and length information in these cases both shall be inferred.

4) As defined in ISO/IEC 14443-3.

5) As defined in ISO/IEC 7816-4.

9.4 Retrieval of a profile

9.4.1 General

A terminal retrieves the profile ID from the card. There are generally four ways a terminal may obtain all the necessary information to enable it to work with a card, product or function these are shown in the following subclauses:

9.4.2 Information via profile ID

The card provides the profile ID. The terminal finds all relevant information in its own stored profile(s) to perform the transaction or function with the card.

9.4.3 Information via profile ID and additional card data

The terminal retrieves the profile ID from the card. Additional data elements on the card override the information in the terminal profile. For example these data elements could be used to change the key version of a card/product without changing the terminal profile.

9.4.4 Information via card data

When a new product is required to be accepted by a terminal it may not be feasible to speedily upgrade all terminals with the adequate profile, especially those operating offline. In such a situation, subject to the card having sufficient memory and security capabilities, the terminal could be provided with the upgrade information from the card.

9.4.5 Information via the IFM

There will be occasions when card transactions shall be referred to the product or application owner before a transaction can be completed. In this event the profile may well be modified either temporarily or permanently by this interaction.

9.5 Terminal profile

Terminal profiles use the two logical interfaces described in Clause 7. One part of the profile is most pertinent to the card and another part of the profile is most pertinent to the data groups. However this split between the two, whilst aiding the understanding of main constituents of a profile, is somewhat arbitrary. In reality the terminal will operate with a single profile that conditions the use of both logical interfaces dependent on the combination of cards, products, use cases and terminal functionalities.

In general a terminal profile consists of different scripts of card related commands conditioned by the requirements for security and of the application.

The structure of a terminal profile is out of scope of this European Standard. Terminal manufacturers will devise their own methodology in implementing scripts.

9.6 Approach

The profile in a terminal can be seen as a multi-layer driver that allows the terminal to work with a card as described in Annex G.

Finding the reference to the profile may be achieved in one of the following ways:

— during anti-collision (for simple memory (SAK byte) with or without AFI);

— at protocol level (ATS (type A) or application bytes (type B));

— at card level (contents of logical sector n);

— at application level (response to select AID).

9.7 Profile information on the card

A card may deliver to the terminal a profile ID only, some parameters of a profile or a complete profile.

Terminal and profile content related data elements are outside the scope of this European Standard as it is the responsibility of the IFM to define and specify these as needed for interoperability. Figure 17 illustrates the interactive nature of profile derivation.

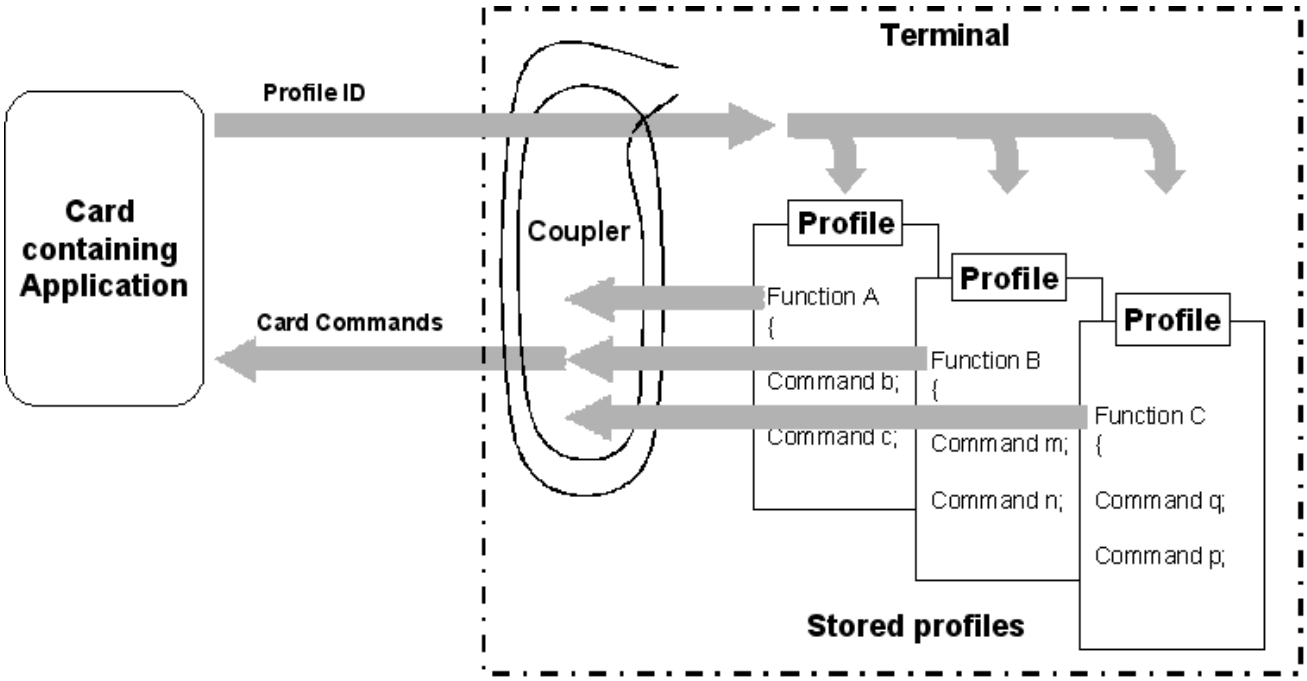

Figure 17 — Profile derivation

Annex A
(normative)

Data group definitions

A.1 Data groups

Table A.1 — Data Group Identification

DATA GROUP	DESCRIPTION
Application environment	Contains fixed and variable data pertaining to the application and default values where the variable data includes a set of labels referring to product data groups and a holder data group (if present) within the application.
Holder	Contains data providing personal and entitlement information concerning the customer of the application.
Event log	The format of each record within the cyclic log reflecting the usage of products. The event log reflects events as they occur and is openly readable by all parties within the Interoperable Fare Management (IFM) system.
Products	Products are described below (see next four table entries).
Charge to Account	This product identifies and provides further information concerning an authorised account which may act as a ticket for a journey or be used to acquire a separate ticket.
Stored Travel Rights	This product identifies and provides further information which may act as a ticket for a journey or be used to acquire a separate ticket for a journey. Stored Travel Rights are often described as a variable length carnet or an undefined ticket.
Entitlement	This product represents a contract between the product owner (via a retail function) and the customer which provides for special rights. An entitlement on its own may not be an authority to travel.
Ticket	This product represents the travel contract which describes the usage, pricing and commercial rules operating for this journey.
Wrapper	Acts as a container for legacy, non-application compliant products to enable them to exist in an application environment. Processing of these products is outside the scope of application and is left to the legacy systems.

Table A.2 — Data structures within data groups

STRUCTURE	DESCRIPTION
ADMINISTRATIVE DATA BLOCKS	**Mandatory administrative data blocks** Required in every data group
Label	The identifier for a data group.
Instance identifier	Contains information relevant to this instance of a data group.
Seal	Contains information which is used to authenticate the data being sealed.
M STRUCTURES	**A series of data objects containing mandatory data elements relevant to the data group. Only one instance of the M data structure may occur in any data group.**
M Application environment	Application environment basic, control and administrative data. Includes administrative data blocks and reference to the labels of other data groups
M event log	Cyclic event log basic, control and administrative data.
M holder	Holder basic, control and administrative data.
M Charge to Account	Charge to Account basic, control and administrative data.
M Stored Travel Rights	Stored Travel Rights basic, control and administrative data.
M customer entilement	Customer entitlement basic, control and administrative data.
M ticket	Ticket basic, control and administrative data.
A STRUCTURES	**Optional data structures usually created at the time the data group is created and usually set to read only access conditions.**
Validity period	A data structure containing boundary dates and/or times.
Retailer	Contains retailer details.
Deposit	Holds a deposit value if present.
Loyalty	Identifies the loyalty scheme.
Holder personal detail	Selected further personal details concerning the holder.
Holder corporate detail	Information about the company which has a contract with the application issuer.
Customer preferences	Default values about the customer stored within the holder group.

Table A.2 (*continued*)

STRUCTURE	DESCRIPTION
CTA additional info	Charge to account, information about the transaction limit.
Stored Travel Rights auto load	Information about threshold (defined by the product owner) at which the autoload facility adds a predetermined value (chosen by the customer in agreement with the product owner).
Ride based travel load	Auto top up parameters for a carnet ticket.
Stored Travel Rights information	Details of Stored Travel Rights.
Customer entitlement detail	Additional entitlement information concerning special rights to which products may apply.
Ticket geography	Information describing the origin and destination of a journey.
Ticket validity information	Information about the validity of tickets and entitlement products.
Ticket restrictions	Information providing additional limitations on ticket and entitlement products.
Reservation basic	Minimum reservation information required.
Reservation additional	Additional reservation information usually required for inspection and validation purposes.
Ticket special conditions	Specific information concerning the ticket applying in different situations.
L STRUCTURES	**Optional data structures usually added at an event after the data group was created and usually set to read/write access conditions.**
Problem log	Information concerning unusual or non-standard events that is to be retained for later analysis.
STR balance	Stored Travel Rights current balance.
STR load	Information concerning the loading of value to an STR account at an event.
STR/CTA usage	Information concerning the usage of STR or CTA at an event.
Ride based ticket usage	Information concerning the use of a carnet ticket at an event.
Ride based ticket balance	The current balance of a carnet ticket.
Ride based ticket load	Information concerning the auto top-up of a carnet ticket at an event.
Loyalty usage	Information concerning loyalty action taken at an event.
Saved event data	Event information to be retained and therefore kept separately from the event log in a relevant data group, usually a product data group.
User specific	A structure containing data not otherwise specified within application and undefined within it. May be used to house EN 1545 data elements not defined for use in the application.

Table A.2 (*continued*)

STRUCTURE	DESCRIPTION
Payment receipt	The electronic receipt for payment. This may map to a physical receipt.
Event log	The structure representing application cyclic event log records.

Annex B
(normative)

Identification and mapping of data groups

B.1 Application environment specific mandatory data structures

In Tables B.1 to B.5, the following is noted:

— annotation M indicates the mandatory presence of the indicated data structure or data element.within a data structure. Where mandatory data elements are indicated within optional data structures, the implication is that if the data structure is present then it shall include all mandatory data elements;

— annotation O indicates optional presence of the indicated data structure or data element within a data structure. Where optional data elements are indicated within a mandatory data structure, the implication is that the data structure shall be present but the optional data elements may or may not be contained within it according to need. The expression of data elements as TLV data objects means that the data structure can be deciphered;

— INTS is a datatype defined in EN 1545.

Table B.1 — Application environment specific mandatory data structures

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	
M ApplicationEnvironment			**M**
applicationEnvironmentLabel		SEQUENCE	M
applicationEnvironmentInstanceIdentifierFixed		SEQUENCE	M
applicationEnvironmentSealFixed		SEQUENCE	M
applicationEnvironmentInstanceIdentifierVariable		SEQUENCE	O
applicationEnvironmentSealVariable		SEQUENCE	O
applicationEnvironmentSamSerialNumber	serialNumber	ReferenceNumber	M
applicationEnvironmentDefaultCurrency	paymentUnit	OCTET STRING	M
applicationEnvironmentMappingType	MappingType	INTS	M
applicationEnvironmentSaleDate	startDate	DateCompact	M
applicationEnvironmentMasterTransactionCounter	transactionSequenceNumber	SequenceNumber	M

Table B.1 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	
M ApplicationEnvironmentLabel			**M**
applicationOwner	companyId	ReferenceNumber	M
expiryDate	endDate	DateCompact	M
M ApplicationEnvironmentInstanceIdentifier [a]			**M**
unblockInstanceNumber	unblockInstanceNumber	SequenceNumber	M
status	productStatus	StatusCode	M
versionNumber	versionNumber	BIT STRING	M
dataGroupSequenceNumber	sequenceNumber	INTEGER	M
security	securityVersion	SEQUENCE	
M Seal [b]			**M**
seal	authenticator	OCTET SRING	M

a Instance identifiers and seals are identified by their unique Tags (representing fixed, or variable) and the seal and instance identifier data structures may be updated with appropriate authority.

b Instance identifiers and seals are identified by their unique Tags (representing fixed, or variable) and the seal and instance identifier data structures may be updated with appropriate authority.

B.2 Event log specific mandatory data structures

Table B.2 — Event log specific mandatory data structures

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	
M Event Log			**M**
eventLogLabel		SEQUENCE	M
eventLogInstanceIdentifierFixed		SEQUENCE	M
eventLogSealFixed		SEQUENCE	M
eventLogInstanceIdentifierVariable		SEQUENCE	O
eventLogSealVariable		SEQUENCE	O
eventLogSamSerialNumber	serialNumber	ReferenceNumber	M
M Event Log Label [a]			**M**
applicationOwner	companyId	ReferenceNumber	M
expiryDate	endDate	DateCompact	M
M Event Log Instance Identifier			M
versionNumber	versionNumber	BIT STRING	M
dataGroupSequenceNumber	sequenceNumber	INTEGER	M
security	securityVersion	SEQUENCE	M
mostRecentPointer	mostRecentPointer	InstancePointer	M
Seal			**M**
seal	authenticator	OCTET STRING	M

a The event log label is repeated in the application environment.

B.3 General mandatory data structures

Table B.3 — General mandatory data structures

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			Application environment	Holder	Charge To Account	Stored Travel Rights	Entitlement	Ticket	Wrapper
M Holder				M					
holderLabel		SEQUENCE	M	M					
holderInstanceIdentifierFixed		SEQUENCE		M					
holderSealFixed		SEQUENCE		M					
holderInstanceIdentifierVariable		SEQUENCE		O					
holderSealVariable		SEQUENCE		O					
holderSamSerialNumber	serialNumber	ReferenceNumber		M					
M Charge To Account					M				
ctaLabel		SEQUENCE	M		M				
ctaInstanceIdentifierFixed		SEQUENCE			M				
ctaSealFixed		SEQUENCE			M				
ctaInstanceIdentifierVariable		SEQUENCE			O				
ctaSealVariable		SEQUENCE			O				
ctaSamSerialNumber	serialNumber	ReferenceNumber			M				

Table B.3 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			Application environment	Holder	Charge To Account	Stored Travel Rights	Entitlement	Ticket	Wrapper
M Stored Travel Rights						M			
strLabel		SEQUENCE	M			M			
strInstanceIdentifierFixed		SEQUENCE				M			
strSealFixed		SEQUENCE				M			
strInstanceIdentifierVariable		SEQUENCE				O			
strSealVariable		SEQUENCE				O			
strSamSerialNumber	serialNumber	ReferenceNumber				M			
strUnit	paymentUnit	OCTET STRING				M			
strMaxValue	maxAmountLimit	Amount				M			
M-Customer Entitlement							M		
entitlementLabel		SEQUENCE	M				M		
entitlementInstanceIdentifierFixed		SEQUENCE					M		
entitlementSealFixed		SEQUENCE					M		
entitlementInstanceIdentifierVariable		SEQUENCE					O		
entitlementSealVariable		SEQUENCE					O		
entitlementSamSerialNumber	serialNumber	ReferenceNumber					M		
entitlementType	entitlementTypeCode	ENUMERATED					M		
entitlementDossierReference	dossierId	ReferenceIdentifier					M		

Table B.3 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			Application environment	Holder	Charge To Account	Stored Travel Rights	Entitlement	Ticket	Wrapper
M Ticket								M	
ticketLabel		SEQUENCE	M					M	
ticketInstanceIdentifierFixed		SEQUENCE						M	
ticketSealFixed		SEQUENCE						M	
ticketInstanceIdentifierVariable		SEQUENCE						O	
ticketSealVariable		SEQUENCE						O	
ticketSamSerialNumber	serialNumber	ReferenceNumber						M	
ticketPrice	price	SEQUENCE						M	
M Wrapper [a]									M
wrapperLabel			M						M
wrapperInstanceIdentifierFixed									M
wrapperSealFixed									M
wraapperInstanceIdentifierVariable									O
wrapperSealVariable									O
wrapperSamSerialNumber	serialNumber	ReferenceNumber							M
wrapperApplicationIdentifier		ReferenceIdentifier							M

Table B.3 (*continued*)

STRUCTURE	EN1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			Application environment	Holder	Charge To Account	Stored Travel Rights	Entitlement	Ticket	Wrapper
M Product Label			M	M	M	M	M	M	M
productOwner	CompanyId	ReferenceNumber	M	M	M	M	M	M	M
expiryDate	EndDate	DateCompact	M	M	M	M	M	M	M
productId	ProductId	ReferenceIdentifier	M	M	M	M	M	M	M
M Product Instance Identifier									
unblockInstanceNumber	unblockInstanceNumber	SequenceNumber		M	M	M	M	M	M
status	productStatus	StatusCode		M	M	M	M	M	M
versionNumber	versionNumber	BIT STRING		M	M	M	M	M	M
dataGroupSequenceNumber	sequenceNumber	INTEGER		M	M	M	M	M	M
security	securityVersion	SEQUENCE		M	M	M	M	M	M
priority	priority	INT1			O	O	O	O	O
retailerIdentifier	NetworkSpecificCompanyID	ReferenceNumber			O	O	O	O	
Seal				M	M	M	M		M
seal	authenticator	OCTET STRING		M	M	M	M		M

a WrapperApplicationIdentifier is allocated by the registration authority.

B.4 Optional data structures – Type A

Table B.4 — Type A optional data structures

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
A-Validity Period			A	A		A	A	A	A
validityStartDate	startDate	DateCompact	O	O		O	O	O	O
validityEndDate	endDate	DateCompact	O	O		O	O	O	O
validityStartTime	startTime	TimeCompact	O	O		O	O	O	O
validityEndTime	endTime	TimeCompact	O	O		O	O	O	O
A-Retailer			A	A		A	A	A	A
retailerIdentifier	NetworkSpecificCompanyID	ReferenceNumber				O	O	O	O
retailerSaleAgent	companyId	ReferenceNumber				O	O	O	O
retailerSaleDevice	deviceId	ReferenceIdentifier				O	O	O	O
A-Deposit			A	A		A	A	A	A
deposit	price	SEQUENCE	O				O		
A-Loyalty			A	A		A	A	A	A
loyaltyMembershipID	membershipId	ReferenceIdentifier				O	O	O	O
loyaltySchemeId	loyaltySchemeId	ReferenceIdentifier				O	O	O	O

Table B.4 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
A-Holder Personal Detail			A	A		A	A	A	A
holderSecondaryFlag	secondaryFlag	Flag		O					
holderSurname	surname	Name		O					
holderForename	forename	Name		O					
holderGender	genderCode	BIT STRING		O					
holderBirthName	birthName	Name		O					
holderBirthDate	birthDate	Datef		O					
holderBirthPlace	birthPlace	Name		O					
holderProfile	ProfileCodeIOP	INTM		O					
holderIdNumber	holderId	ReferenceIdentifier		O					
holderCountryAlpha	countryAlpha	PrintabelString		O					
holderAddress	holderAddress	Address		O					
HolderZipcode	postCodeId	ReferenceIdentifier		O					
holderTelephone	telephone	NetworkAccess		O					
holderMobile	telephone	NetworkAccess		O					
holderFax	fax	NetworkAccess		O					
holderEmail	emailAddress	NetworkAccess		O					
holderLanguage	languageId	CHOICE		O					
holderIdDocumentNumber	identityDocumentId	ReferenceIdentifier		O					

Tables B.4 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
A Holder Corporate Detail			A	A		A	A	A	A
holderCompanyName	companyName	Name		O					
holderCompanyAddress	holderAddress	Address		O					
holderCompanyZipcode	postcodeId	ReferenceIdentifier		O					
holderCompanyTelephone	telephone	NetworkAccess		O					
A Customer Preferences			A	A		A	A	A	A
preferencePassengerClass	accommodationClassCode	ENUMERATED		O					
preferenceSmoker	smoking	Flag		O					
preferencePaymentMeans	paymentMeansCode	BIT STRING		O					
preferenceDirection	directionCode	ENUMERATED		O					
A-CTA Additional Info			A	A		A	A	A	A
ctaTransactionLimit	price	SEQUENCE				O			
ctaAccountingReference	accountingId	ReferenceIdentifer				O			
A-Stored Travel Rights Auto Load			A	A		A	A	A	A
strAutoLoadThreshold	thresholdAmount	SignedAmount					O		
strAutoLoadValue	maxAmountLimit	Amount					O		

Table B.4 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
A-Ride Based Travel Load			A	A		A	A	A	A
rbtMaxLoadLimit	maxAmountLimit	Amount							O
rbtMaxLoadValue	maxAmountLimit	Amount							O
rbtValue	maxAmountLimit	Amount							O
rbtUnitType	travelServiceId	ReferenceIdentifer							O
rbtReuseLockTime	lockTime	INT1							O
A-Stored Travel Rights Information			A	A		A	A	A	A
strSerialNumber	serialNumber	ReferenceNumber					O		
strMaxTransactionValue	maxAmountLimit	Amount					O		
strMinTransactionValue	minAmountLimit	SignedAmount					O		
strReloadSelection	strLoadCode	ENUMERATED					O		
A-Customer Entitlement Detail			A	A		A	A	A	A
EntitlementClass [a]	accommodationClassCode	ENUMERATED		O				O	
entitlementType	entitlementTypeCode	ENUMERATED		O				O	
entitlementPassengerTotal	passengerTotal	Quantity		O				O	
entitlementSerialNumber	serialNumber	ReferenceNumber		O				O	

Table B.4 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
entitlementPrice	price	SEQUENCE		O				O	
entitlementReference	accountingId	ReferenceIdentifier		O				O	
enitlementDiscount	discountCode	INTM		O				O	
A-Ticket Geography			A	A		A	A	A	A
ticketJourneyOrigin	origin	LocationIdentifier							O
ticketJourneyDestination	destination	LocationIdentifier							O
ticketZoneMap	zoneMap	INT3							O
ticketZoneId	zoneId	ReferenceIdentifier							O
ticketText	userData	Databin							0
A Ticket Validity Information			A	A		A	A	A	A
validityRemoveDate	date	DateCompact						O	O
validitySaleDateTimeStamp	dateTimeStamp	INT3						O	O
validityDepartureDateTimeStamp	dateTimeStamp	INT3						O	O
ValidityPassengerClass [b]	accommodationClassCode	ENUMERATED						O	O
validityAccompaniedBy	accompaniedBy	SEQUENCE						O	O
validityNumberofAdults	numberOfAdults	Quantity						O	O

Table B.4 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
validityNumberofChildren	numberOfChildren	Quantity						O	O
validityPaymentMeans	paymentMeansCode	BIT STRING						O	O
validityTariffNumber	tariffNumber	ReferenceNumber						O	O
validitySerialNumber	serialNumber	ReferenceNumber						O	O
validityTrainCategory	transportTypeCode	INTP						O	O
validityNotVia	locationIdentifier	SEQUENCE						O	O
validityContractDependencyPointer	contractDependencyPointer	InstancePointer						O	O
validityValidationModel	validationModelCode	ENUMERATED						O	O
validityText	userData	Databin						O	O
A Ticket Restrictions			A	A		A	A	A	A
ticketJourneyInterchanges	interchangesAllowed	Quantity		O				O	O
ticketJourneyDistance	journeyDistance	Length		O				O	O
TicketJourneyOrigin	locationIdentifier	SEQUENCE		O				O	O
ticketJourneyDestination	locationIdentifier	SEQUENCE		O				O	O
ticketJourneyRun	journeyRunId	ReferenceNumber		O				O	O
ticketJourneyVia	locationIdentifier	SEQUENCE		O				O	O
ticketPeriodJourneys	periodJourneys	Quantity		O				O	O
ticketRestriction	restriction	INT2		O				O	O

Table B.4 (*continued*)

STRUCTURE	EN1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
ticketRestrictEnd	restrictionEnd	DateTimeStamp		O				O	O
ticketRestrictLocation	locationIdentifier	SEQUENCE		O				O	O
ticketRestrictStart	restrictionStart	DateTimeStamp		O				O	O
ticketRestrictedPeriodOf Day	restrictedPeriodOfDay	PeriodOfDay		O				O	O
ticketServices	transportTypeCode	INTP		O				O	O
ticketValidityJourneys	countOfJourneys	Counter		O				O	O
ticketValidityLimitDate	date	DateCompact		O				O	O
ticketVehicleClassAllowed	accommodationClassCode	ENUMERATED		O				O	O
ticketFurthestPlace	locationIdentifier	SEQUENCE		O				O	O
ticketEndTimeStamp	endTimeStamp	TimeStamp		O				O	O
ticketNumberOfPassbacks	numberOfPassbacks	Quantity		O				O	O
ticketPassbackTime	passbackTime	Quantity		O				O	O
ticketRestrictDOW	restrictedDayOfWeek	DayOfWeek		O				O	O
ticketZoneId	zoneId	ReferenceIdentifier		O				O	O
A Ticket Service Validity			A	A		A	A	A	A
ticketJourneyRoute	routeId	ReferenceNumber						O	O
ticketJourneyRouteVariant	routeVariantId	ReferenceNumber						O	O

Table B.4 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
A Reservation Basic [c]			A	A		A	A	A	A
reservationServiceId	travelServiceId	ReferenceIdentifier							O
ReservationDepartureDateTimeStamp	dateTimeStamp	INT3							O
reservationCoachIdFrom	reservationReferenceId	ReferenceNumber							O
reservationCoachIdTo	reservationReferenceId	ReferenceNumber							O
ReservationCoachAlphaFrom	seatAlphaId	ReferenceIdentifier							O
reservationCoachAlphaTo	seatAlphaId	ReferenceIdentifier							O
reservationSeatNumberFrom	seatNumber	ReferenceNumber							O
reservationSeatNumberTo	seatNumber	ReferenceNumber							O
reservationSeatAlphaFrom	seatAlphaId	OCTET STRING							O
reservationSeatAlphaTo	seatAlphaId	OCTET STRING							O
reservationPlaceType	seatPositionCode	ENUMERATED							O
A Reservation Additional			A	A		A	A	A	A
reservationNumber	reservationReferenceId	ReferenceNumber							O
reservationOverbookingIndicator	overbookingIndicator	Flag							O
reservationPhysicalClass	accommodationClassCode	ENUMERATED							O
reservationServiceOperator	companyId	ReferenceNumber							O
reservationSmokers	smoking	Flag							O

Table B.4 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	DATA GROUP						
			ENVIRONMENT	HOLDER		CHARGE TO ACCOUNT	STORED TRAVEL RIGHTS	ENTITLEMENT	TICKET
reservationSpecialFacility	assistanceTypeCode	ENUMERATED							O
reservationArrivalDateTimeStamp	dateTimeStamp	INT3							O
reservationDossierReference	dossierId	ReferenceIdentifier							O
reservationService	extraServiceCode	INTM							O
reservationOrigin	locationIdentifier	SEQUENCE							O
reservationDestination	locationIdentifier	SEQUENCE							O
A Ticket Special Conditions			A	A		A	A	A	A
ticketLastMinuteSale	lastMinuteSale	Flag							O
ticketLegislativeBackground	legislationCode	ENUMERATED							O
A Blocking Status			A	A		A	A	A	A
blockingDateTimeStamp	dateTimeStamp	INT3	O	O		O	O	O	O
blockingReason	hotListStatusCode	ENUMERATED	O	O		O	O	O	O
blockingSamSerialNumber	serialNumber	ReferenceNumber	O	O		O	O	O	O

a Refers to the passenger's entitlement rather than what is actually allocated which is shown in validityPassengerClass.

b Refers to what is actually allocated rather than the passenger's entitlement which is shown in entitlementClass.

c May be repeated if multiple coaches or seats are reserved. Where only one coach or seat is reserved, the appropriate 'From' data element is used and the 'To' data element is omitted

B.5 Optional data structures – Type L

Table B.5 — Type L data structures

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	Data Group						
			Environment	Holder		Charge To Account	Stored Travel Rights	Entitlement	Ticket
L Problem Log			L	L		L	L	L	L
problemDateTimeStamp	dateTimeStamp	INT3	O						
problemEventCode	eventTypeCode	ENUMERATED	O						
problemSamSerialNumber	serialNumber	ReferenceNumber	O						
problemDevice	deviceId	ReferenceIdentifier	O						
problemDeviceSerialNumber	serialNumber	ReferenceNumber	O						
problemResult	resultCode	BIT STRING	O						
problemNotOKCounter	notOKCounter	Counter	O						
problemSeriousness	seriousnessCode	ENUMERATED	O						
L Stored Travel Rights Balance									
strValue	balance	SignedAmount					O		
L Stored Travel Rights Load									
strDateTimeStamp	dateTimeStamp	INT3					O		
strSequenceNumber	transactionSequenceNumber	SequenceNumber					O		
strSamSerialNumber	serialNumber	ReferenceNumber					O		
strRetailerId	companyId	ReferenceNumber					O		

Table B.5 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	Data Group						
			Environment	Holder		Charge To Account	Stored Travel Rights	Entitlement	Ticket
strAmountLoaded	maxAmountLimit	Amount					O		
strAfterLoadBalance	balance	SignedAmount					O		
strReceiptFlag	receiptPrintedFlag	Flag					O		
L STR/CTA Usage			**L**	**L**		**L**	**L**	**L**	**L**
(str/cta)FareDeducted (notional in CTA)	fareDeducted	Amount				O	O		
(str/cta)PriceReduction	discountCode	INTM				O	O		
(str/cta)CumulativeFare	cumulativeFare	Amount				O	O		
(str/cta)DateOfLastRide	date	DateCompact				O	O		
(str/cta)TimeOfLastRide	time	TimeCompact				O	O		
(str/cta)CountOfJourneys	countOfJourneys	Counter				O	O		
(str/cta)ServiceOfLastRide	travelServiceId	ReferenceIdentifier				O	O		
(str/cta)LocationIdentifier	locationIdentifier	SEQUENCE				O	O		
(str/cta)ContractDependencyPointer	sequenceNumber	INTEGER				O	O		
L Ride Based Travel Usage									
rbtCouponsDeducted	couponsDeducted	Quantity							O
rbtCountOfCoupons	countOfCoupons	Counter							O

Table B.5 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	Data Group						
			Environment	Holder		Charge To Account	Stored Travel Rights	Entitlement	Ticket
L Ride Based Travel Load									
rbtDateTimeStamp	dateTimeStamp	INT3							O
rbtSamSerialNumber	serialNumber	ReferenceNumber							O
rbtRetailerIdentifier	companyId	ReferenceNumber							O
rbtCouponsLoaded	couponsLoaded	Quantity							O
rbtLoadPrice	price	SEQUENCE							O
rbtAfterLoadBalance	balance	SignedAmount							O
L Loyalty Balance									
loyaltyPoints	loyaltyPoints	Counter				O	O	O	O
loyaltyValidityDuration	validityDuration	Duration				O	O	O	O
loyaltySchemeId	loyaltySchemeId	ReferenceIdentifier				O	O	O	O
L Primary Saved Event Data (n)			**L**	**L**		**L**	**L**	**L**	**L**
savedEventTypeCode	eventTypeCode	ENUMERATED	O	O		O	O	O	O
savedEventSequenceNumber	transactionSequenceNumber	SequenceNumber	O	O		O	O	O	O
savedEventDateTimeStamp	dateTimeStamp	INT3	O	O		O	O	O	O
savedEventResultCode	resultCode	BIT STRING	O	O		O	O	O	O
savedEventDeviceSerialNumber	serialNumber	ReferenceNumber	O	O		O	O	O	O
savedEventDeviceTypeCode	deviceTypeCode	ENUMERATED	O	O		O	O	O	O

Table B.5 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	Data Group						
			Environment	Holder		Charge To Account	Stored Travel Rights	Entitlement	Ticket
savedEventServiceOperator	companyId	ReferenceNumber	O	O		O	O	O	O
savedEventJourneyOrigin	locationIdentifier	SEQUENCE	O	O		O	O	O	O
savedEventJourneyDestination	locationIdentifier	SEQUENCE	O	O		O	O	O	O
L Secondary Saved Event Data (n)			**L**	**L**		**L**	**L**	**L**	**L**
savedEventStatusReceiptFlag	receiptPrintedFlag	Flag	O	O		O	O	O	O
savedEventDeviceOwner	companyId	ReferenceNumber	O	O		O	O	O	O
savedEventDevice	deviceId	ReferenceIdentifier	O	O		O	O	O	O
savedEventDeviceSequenceNumber	sequenceNumber	INTEGER	O	O		O	O	O	O
savedEventJourneyVia	locationIdentifier	SEQUENCE	O	O		O	O	O	O
savedEventJourneyRoute	routeId	ReferenceNumber	O	O		O	O	O	O
savedEventJourneyRouteVariant	routeVariantId	ReferenceNumber	O	O		O	O	O	O
savedEventJourneyRun	journeyRunId	ReferenceNumber	O	O		O	O	O	O
savedEventDirection	directionCode	ENUMERATED	O	O		O	O	O	O
savedEventCountOfJourneys	countOfJourneys	Counter	O	O		O	O	O	O
savedEventTotalJourneys	countOfJourneys	Counter	O	O		O	O	O	O
savedEventTicketUseValidityDate	endDate	DateCompact	O	O		O	O	O	O
SavedEventTicketUseValidityTime	endTime	TimeCompact	O	O		O	O	O	O
savedEventJourneyDistance	journeyDistance	Length	O	O		O	O	O	O
savedEventJourneyInterchanges	interchangesAllowed	Quantity	O	O		O	O	O	O

Table B.5 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	Data Group						
			Environment	Holder		Charge To Account	Stored Travel Rights	Entitlement	Ticket
savedEventZoneId	zoneId	ReferenceIdentifier	O	O		O	O	O	O
savedEventPricePaid	price	SEQUENCE	O	O		O	O	O	O
savedEventPaymentMeans	paymentMeansCode	BIT STRING	O	O		O	O	O	O
savedEventNonDiscountedPrice	price	SEQUENCE	O	O		O	O	O	O
savedEventValidationCounter	validationCounter	Counter	O	O		O	O	O	O
savedEventValidationStatus	productStatus	StatusCode	O	O		O	O	O	O
savedEventTransportServiceCode	transportTypeCode	INTP	O	O		O	O	O	O
savedEventVehicleClass	accommodationClassCode	ENUMERATED	O	O		O	O	O	O
savedEventVehicleId	vehicleId	ReferenceIdentifier	O	O		O	O	O	O
savedEventLocationIdentifier	locationIdentifier	SEQUENCE	O	O		O	O	O	O
savedEventPassengerClass	accommodationClassCode	ENUMERATED	O	O		O	O	O	O
savedEventCustomerProfile	ProfileCodeIOP	INTM	O	O		O	O	O	O
savedEventPassengerTotal	passengerTotal	Quantity	O	O		O	O	O	O
savedEventText	userData	Databin	O	O		O	O	O	O
L Zone List			**L**	**L**		**L**	**L**	**L**	**L**
ZoneListZoneId	zoneId	ReferenceIdentifier		O		O	O	O	O
(repeated up to 40 entries)									

Table B.5 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	Data Group						
			Environment	Holder		Charge To Account	Stored Travel Rights	Entitlement	Ticket
L User Specific			**L**	**L**		**L**	**L**	**L**	**L**
userDataOwner	companyId	ReferenceNumber	O	O		O	O	O	O
userData	userData	Databin	O	O		O	O	O	O
L Payment Receipt			**L**	**L**		**L**	**L**	**L**	**L**
paymentCertificateAccountNumber	accountNumber	Iai				O	O		
paymentCertificatePaymentMeans	paymentMeansCode	BIT STRING				O	O		
paymentCertificatePaymentProvider	companyId	ReferenceNumber				O	O		
paymentCertificateData	userData	Databin				O	O		
paymentCertificatePrice	price	SEQUENCE				O	O		
paymentCertificateStrBalance	balance	SignedAmount				O	O		
paymentReceiptAccountNumber	accountNumber	Iai				O	O		
paymentReceiptData	receiptData	Databin				O	O		
paymentReceiptDate	date	DateCompact				O	O		
paymentReceiptDevice	deviceId	ReferenceNumber				O	O		
paymentReceiptLocationIdentifier	locationIdentifier	SEQUENCE				O	O		
paymentReceiptPaymentMeans	paymentMeansCode	BIT STRING				O	O		
paymentReceiptPaymentProvider	companyId	ReferenceNumber				O	O		
paymentReceiptPrice	price	SEQUENCE				O	O		
paymentReceiptStrBalance	balance	SignedAmount				O	O		

Table B.5 (*continued*)

STRUCTURE	EN 1545 DATA ELEMENT	DATA TYPE	Data Group						
			Environment	Holder		Charge To Account	Stored Travel Rights	Entitlement	Ticket
paymentReceiptResult	resultCode	BIT STRING				O	O		
paymentReceiptSerialNumber	serialNumber	ReferenceNumber				O	O		
paymentReceiptServiceOperator	companyId	ReferenceNumber				O	O		
PaymentPayServicePoint	transportTypeCode	INTP				O	O		
paymentPayServicePointInfo	userActionCode	INTP				O	O		

B.6 Cyclic event log record data structure

Table B.6 — Cyclic event log data structure

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATATYPE
eventTypeCode	eventTypeCode	ENUMERATED
eventSequenceNumber	transactionSequenceNumber	SequenceNumber
eventContractDependencyPointer	sequenceNumber	INTEGER
eventDateTimestamp	dateTimeStamp	INT3
eventPassbackTime	passbackTime	Quantity
eventResultCode	resultCode	BIT STRING
eventSamSerialNumber	serialNumber	ReferenceNumber
eventDeviceId	deviceId	ReferenceIdentifier

B.7 General event log

This log file may or may not exist and it is in any event outside the scope of the application and outside this European Standard. It may be used as desired:

— this log would normally be created as a rotating or cyclic file at the card level;

— although outside the application in both use and location on the card, application data may be logged here without restriction;

— this log will most often be used for diagnostic purposes.

Annex C
(normative)

EN 1545 data elements enumerated for use in the application

C.1 Relationship between the application and EN 1545

EN 1545 defines the data elements that are involved in surface transport card data. In EN 1545, these data elements are listed, defined to explain their use and context, and structured in terms of their format including codification and/or sizing constraints where relevant using ASN.1 notation.

The specification of data elements in EN 1545 implies a logical definition of data elements in which the actual format and size of the data elements is defined by the implementer and the encoding rule used. In particular, EN 1545 makes extensive use of the enumerated type for code lists; as well as the range and choice options for other data elements. In the application, to achieve interoperability, noting that it is likely that PER or similar encoding rules will be applied to minimise the card data set implemented in real environments, it is necessary that some bounds be applied to these flexible fields.

C.2 Application data elements fully specified in EN 1545

Table C.1 — Application data elements fully specified in EN 1545

Application Usage	EN 1545 Data Element	Data Type
(str/cta)DateOfLastRide	Date	DateCompact
(str/cta)PriceReduction	DiscountCode	INTM
(str/cta)TimeOfLastRide	Time	TimeCompact
ApplicationEnvironmentDefaultCurrency	PaymentUnit	OCTET STRING
ApplicationEnvironmentMappingType	MappingType	INTS
ApplicationEnvironmentSaleDate	StartDate	DateCompact
BlockingDateTimeStamp	DateTimeStamp	INT3
BlockingReason	HotListStatusCode	ENUMERATED
EntitlementClass	AccommodationClassCode	ENUMERATED
EntitlementDiscount	DiscountCode	INTM
EntitlementType	EntitlementTypeCode	ENUMERATED
EventDateTimestamp	DateTimeStamp	INT3
EventResultCode	ResultCode	BIT STRING
EventTypeCode	EventTypeCode	ENUMERATED

Table C.1 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE
ExpiryDate	EndDate	DateCompact
HolderAddress	HolderAddress	Address
HolderBirthDate	BirthDate	Datef
HolderBirthName	BirthName	Name
HolderBirthPlace	BirthPlace	Name
HolderCompanyAddress	HolderAddress	Address
HolderCompanyName	CompanyName	Name
HolderCountryAlpha	CountryAlpha	PrintabelString
HolderForename	Forename	Name
HolderGender	GenderCode	BIT STRING
HolderLanguage	LanguageId	CHOICE
HolderProfile	ProfileCodeIOP	INTM
HolderSecondaryFlag	SecondaryFlag	Flag
HolderSurname	Surname	Name
PaymentCertificateAccountNumber	AccountNumber	Iai
PaymentCertificatePaymentMeans	PaymentMeansCode	BIT STRING
PaymentPayServicePoint	TransportTypeCode	INTP
PaymentPayServicePointInfo	UserActionCode	INTP
PaymentReceiptAccountNumber	AccountNumber	Iai
PaymentReceiptDate	Date	DateCompact
PaymentReceiptPaymentMeans	PaymentMeansCode	BIT STRING
PaymentReceiptResult	ResultCode	BIT STRING
PreferenceDirection	DirectionCode	ENUMERATED
PreferencePassengerClass	AccommodationClassCode	ENUMERATED
PreferencePaymentMeans	PaymentMeansCode	BIT STRING

Table C.1 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE
PreferenceSmoker	Smoking	Flag
Priority	Priority	INT1
ProblemDateTimeStamp	DateTimeStamp	INT3
ProblemEventCode	EventTypeCode	ENUMERATED
ProblemResult	ResultCode	BIT STRING
ProblemSeriousness	SeriousnessCode	ENUMERATED
RbtDateTimeStamp	DateTimeStamp	INT3
RbtReuseLockTime	LockTime	INT1
ReservationArrivalDateTimeStamp	DateTimeStamp	INT3
ReservationDepartureDateTimeStamp	DateTimeStamp	INT3
ReservationOverbookingIndicator	OverbookingIndicator	Flag
ReservationPhysicalClass	AccommodationClassCode	ENUMERATED
ReservationPlaceType	SeatPositionCode	ENUMERATED
ReservationService	ExtraServiceCode	INTM
ReservationSmokers	Smoking	Flag
ReservationSpecialFacility	AssistanceTypeCode	ENUMERATED
SavedEventCustomerProfile	ProfileCodeIOP	INTM
SavedEventDateTimeStamp	DateTimeStamp	INT3
SavedEventDeviceTypeCode	DeviceTypeCode	ENUMERATED
SavedEventDirection	DirectionCode	ENUMERATED
SavedEventPassengerClass	AccommodationClassCode	ENUMERATED
SavedEventPaymentMeans	PaymentMeansCode	BIT STRING
SavedEventResultCode	ResultCode	BIT STRING
SavedEventStatusReceiptFlag	ReceiptPrintedFlag	Flag
SavedEventTicketUseValidityDate	EndDate	DateCompact

Table C.1 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE
SavedEventTicketUseValidityTime	EndTime	TimeCompact
SavedEventTransportServiceCode	TransportTypeCode	INTP
SavedEventTypeCode	EventTypeCode	ENUMERATED
SavedEventValidationStatus	ProductStatus	StatusCode
SavedEventVehicleClass	AccommodationClassCode	ENUMERATED
Status	ProductStatus	StatusCode
Status	ProductStatus	StatusCode
StrDateTimeStamp	DateTimeStamp	INT3
StrReceiptFlag	ReceiptPrintedFlag	Flag
StrReloadSelection	StrLoadCode	ENUMERATED
StrUnit	PaymentUnit	OCTET STRING
TicketEndTimeStamp	EndTimeStamp	TimeStamp
TicketLastMinuteSale	LastMinuteSale	Flag
TicketLegislativeBackground	LegislationCode	ENUMERATED
TicketRestrictDOW	RestrictedDayOfWeek	DayOfWeek
TicketRestriction	Restriction	INT2
TicketServices	TransportTypeCode	INTP
TicketValidityLimitDate	Date	DateCompact
TicketVehicleClassAllowed	AccommodationClassCode	ENUMERATED
TicketZoneMap	ZoneMap	INT3

Table C.1 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE
ValidityDepartureDateTimeStamp	DateTimeStamp	INT3
ValidityEndDate	EndDate	DateCompact
ValidityEndTime	EndTime	TimeCompact
ValidityPassengerClass	AccommodationClassCode	ENUMERATED
ValidityPaymentMeans	PaymentMeansCode	BIT STRING
ValidityRemoveDate	Date	DateCompact
ValiditySaleDateTimeStamp	DateTimeStamp	INT3
ValidityStartDate	StartDate	DateCompact
ValidityStartTime	StartTime	TimeCompact
ValidityTrainCategory	TransportTypeCode	INTP
ValidityValidationModel	ValidationModelCode	ENUMERATED
VersionNumber	VersionNumber	BIT STRING

C.3 Application data elements not fully specified in EN 1545

The appliation data elements in Table C.2 below make use of EN 1545 data elements that are not fully specified in EN 1545 and are enumerated below. The values below show maximum sizes and ranges for data element enumeration. Specific implementations may impose further restrictions.

NOTE The sizing for amount and value reflects the possibility that their expression may be in minor units (e.g. Cents) or in major units (e.g. Euros).

The revision within a release of a version may reflect changes to code lists used within the data structure to which the version number applies.

Table C.2 — Application data elements not fully specified in EN 1545

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE	SPECIFICATION
(str/cta)ContractDependencyPointer	SequenceNumber	INTEGER	0..4,294,967,295
(str/cta)CountOfJourneys	CountOfJourneys	Counter	0..65,535
(str/cta)CumulativeFare	CumulativeFare	Amount	0..16,777,215
(str/cta)FareDeducted (notional in CTA)	FareDeducted	Amount	0..16,777,215
(str/cta)LocationIdentifier	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets

Table C.2 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE	SPECIFICATION
(str/cta)ServiceOfLastRide	TravelServiceId	ReferenceIdentifer	0..65,535
ApplicationEnvironmentMasterTransactionCounter	TransactionSequenceNumber	SequenceNumber	0..4,294,967,295
ApplicationEnvironmentSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
ApplicationOwner	CompanyId	ReferenceNumber	0..65,535
BlockingSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
CtaAccountingReference	AccountingId	ReferenceIdentifer	10 octets
CtaSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
CtaTransactionLimit	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
DataGroupSequenceNumber	SequenceNumber	INTEGER	0..4,294,967,295
DataGroupSequenceNumber	SequenceNumber	INTEGER	0..4,294,967,295
Deposit	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
EntitlementDossierReference	DossierId	ReferenceIdentifer	20 Octets
EntitlementPassengerTotal	PassengerTotal	Quantity	0..255
EntitlementPrice	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
EntitlementReference	AccountingId	ReferenceIdentifer	10 octets
EntitlementSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
EntitlementSerialNumber	SerialNumber	ReferenceNumber	0..65,535
EventContractDependencyPointer	SequenceNumber	INTEGER	0..4,294,967,295
EventDeviceId	DeviceId	ReferenceIdentifier	0..16,777,215
EventLogSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
EventPassbackTime	PassbackTime	Quantity	0..65,535
EventSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
EventSequenceNumber	TransactionSequenceNumber	SequenceNumber	0..4,294,967,295
HolderCompanyTelephone	Telephone	NetworkAccess	18 octets
HolderCompanyZipcode	PostcodeId	ReferenceIdentifer	10 octets
HolderEmail	EmailAddress	NetworkAccess	50 octets
HolderFax	Fax	NetworkAccess	18 octets
HolderIdDocumentNumber	IdentityDocumentId	ReferenceIdentifer	20 octets
HolderIdNumber	HolderId	ReferenceIdentifer	0..4,294,967,215
HolderMobile	Telephone	NetworkAccess	18 octets

Table C.2 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE	SPECIFICATION
HolderSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
HolderTelephone	Telephone	NetworkAccess	18 octets
HolderZipcode	PostCodeId	ReferenceIdentifer	10 octets
LoyaltyMembershipID	LoyaltyMembershipId	ReferenceIdentifer	10 octets
LoyaltyPoints	LoyaltyPoints	Counter	0..16,777,215
LoyaltySchemeId	LoyaltySchemeId	ReferenceIdentifer	5 octets
LoyaltyValidityDuration	ValidityDuration	Duration	Type 0-7 Unit 0-7 0..65,535
MostRecentPointer	MostRecentPointer	InstancePointer	0..255
PaymentCertificateData	UserData	Databin	40 octets
PaymentCertificatePaymentProvider	CompanyId	ReferenceNumber	0..65,535
PaymentCertificatePrice	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
PaymentCertificateStrBalance	Balance	SignedAmount	-32,767..+32,767
PaymentReceiptData	ReceiptData	Databin	40 octets
PaymentReceiptDevice	DeviceId	ReferenceNumber	0..65,535
PaymentReceiptLocationIdentifier	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
PaymentReceiptPaymentProvider	CompanyId	ReferenceNumber	0..65,535
PaymentReceiptPrice	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
PaymentReceiptSerialNumber	SerialNumber	ReferenceNumber	0..65,535
PaymentReceiptServiceOperator	CompanyId	ReferenceNumber	0..65,535
PaymentReceiptStrBalance	Balance	SignedAmount	-32,767..+32,767
ProblemDevice	DeviceId	ReferenceIdentifier	2 octets
ProblemDeviceSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
ProblemNotOKCounter	NotOKCounter	Counter	0..255
ProblemSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
ProductId	ProductId	ReferenceNumber	0..65,535
ProductOwner	CompanyId	ReferenceNumber	0..65,535
RbtAfterLoadBalance	Balance	SignedAmount	-32,767..+32,767
RbtCountOfCoupons	CountOfCoupons	Counter	0..65,535
RbtCouponsDeducted	CouponsDeducted	Quantity	0..65,535

Table C.2 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE	SPECIFICATION
RbtCouponsLoaded	CouponsLoaded	Quantity	0..65,535
RbtLoadPrice	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
RbtMaxLoadLimit	MaxAmountLimit	Amount	0..16,777,215
RbtMaxLoadValue	MaxAmountLimit	Amount	0..16,777,215
RbtRetailerIdentifier	CompanyId	ReferenceNumber	0..65,535
RbtSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
RbtUnitType	TravelServiceId	ReferenceIdentifer	0..65,535
RbtValue	MaxAmountLimit	Amount	0..16,777,215
ReservationCoachAlpha		ReferenceIdentifer	4 octets
ReservationCoachId	ReservationReferenceId	ReferenceNumber	0..65,535
ReservationDestination	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
ReservationDossierReference	DossierId	ReferenceIdentifer	20 octets
ReservationNumber	ReservationReferenceId	ReferenceNumber	0..4,294,97,295
ReservationOrigin	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
ReservationSeatAlphaFrom	SeatAlpha	OCTET SRING	1 octet
ReservationSeatAlphaTo	SeatAlpha	OCTET SRING	1 octet
ReservationSeatNumberFrom	SeatNumber	ReferenceNumber	0..65,535
ReservationSeatNumberTo	SeatNumber	ReferenceNumber	0..65,535
ReservationServiceId	TravelServiceId	ReferenceIdentifer	0..65,535
ReservationServiceOperator	CompanyId	ReferenceNumber	0..65,535
RetailerIdentifier	CompanyId	ReferenceNumber	0..65,535
RetailerSaleAgent	CompanyId	ReferenceNumber	0..65,535
RetailerSaleDevice	DeviceId	ReferenceIdentifier	0..65,535
SavedEventCountOfJourneys	CountOfJourneys	Counter	0..65,535
SavedEventDevice	DeviceId	ReferenceIdentifier	0..65,535
SavedEventDeviceOwner	CompanyId	ReferenceNumber	0..65,535
SavedEventDeviceSequenceNumber	SequenceNumber	INTEGER	0..4,294,967,295
SavedEventDeviceSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295

Table C.2 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE	SPECIFICATION
SavedEventJourneyDestination	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
SavedEventJourneyDistance	JourneyDistance	Length	Type EN 1545 Unit EN 1545 Value 0..65,535
SavedEventJourneyInterchanges	InterchangesAllowed	Quantity	0..255
SavedEventJourneyOrigin	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
SavedEventJourneyRoute	RouteId	ReferenceNumber	0..65,535
SavedEventJourneyRouteVariant	RouteVariantId	ReferenceNumber	0..255
SavedEventJourneyRun	JourneyRunId	ReferenceNumber	0..16,777,215
SavedEventJourneyVia	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
SavedEventLocationIdentifier	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
SavedEventNonDiscountedPrice	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
SavedEventPassengerTotal	PassengerTotal	Quantity	0..255
SavedEventPricePaid	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545
SavedEventSequenceNumber	TransactionSequenceNumber	SequenceNumber	0..65,535
SavedEventServiceOperator	CompanyId	ReferenceNumber	0..65,535
SavedEventText	UserData	Databin	40 octets
SavedEventTotalJourneys	CountOfJourneys	Counter	0..65,535
SavedEventValidationCounter	ValidationCounter	Counter	0..255
SavedEventVehicleId	VehicleId	ReferenceIdentifer	4 octets
SavedEventZoneId	ZoneId	ReferenceIdentifer	2 octets
Seal	Authenticator	OCTET STRING	16 octets
Security	SecurityVersion	SEQUENCE	Algorithm 0..7 Keyvesion 0..7
ServiceReceiptJourneyDistance	JourneyDistance	Length	Type EN 1545 Unit EN 1545 Value 0..65,535
ServiceReceiptSerialNumber	SerialNumber	ReferenceNumber	0..65,535
StrAfterLoadBalance	Balance	SignedAmount	-32,767..+32,767
StrAmountLoaded	MaxAmountLimit	Amount	0..16,777,215

Table C.2 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE	SPECIFICATION
StrAutoLoadThreshold	ThresholdAmount	SignedAmount	-32,767..+32,767
StrAutoLoadValue	MaxAmountLimit	Amount	0..16,777,215
StrMaxTransactionValue	MaxAmountLimit	Amount	0..16,777,215
StrMaxValue	MaxAmountLimit	Amount	0..16,777,215
StrMinTransactionValue	MinAmountLimit	SignedAmount	-32,767..+32,767
StrRetailerId	CompanyId	ReferenceNumber	0..65,535
StrSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
StrSequenceNumber	TransactionSequenceNumber	SequenceNumber	0..4,294,967,295
StrSerialNumber	SerialNumber	ReferenceNumber	0..16,777,215
StrValue	Balance	SignedAmount	-32,767..+32,767
TicketFurthestPlace	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
TicketJourneyDestination	Destination	LocationIdentifier	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
TicketJourneyDestination	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
TicketJourneyDistance	JourneyDistance	Length	Type EN 1545 Unit EN 1545 Value 0..65,535
TicketJourneyInterchanges	InterchangesAllowed	Quantity	0..255
TicketJourneyOrigin	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
TicketJourneyOrigin	Origin	LocationIdentifier	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
TicketJourneyRoute	RouteId	ReferenceNumber	0..65,535
TicketJourneyRouteVariant	RouteVariantId	ReferenceNumber	0..255
TicketJourneyRun	JourneyRunId	ReferenceNumber	0..16,777,215
TicketJourneyVia	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
TicketNumberOfPassbacks	NumberOfPassbacks	Quantity	0..255
TicketPassbackTime	PassbackTime	Quantity	0..65,535
TicketPeriodJourneys	PeriodJourneys	Quantity	0..255
TicketPrice	Price	SEQUENCE	Amount 0..65,535 VAT (amount or %) 0..65,535 Unit EN 1545

Table C.2 (*continued*)

APPLICATION USAGE	EN 1545 DATA ELEMENT	DATA TYPE	SPECIFICATION
TicketRestrictedPeriodofDay	RestrictedPeriodofDay	PeriodOfDay	0..255
TicketRestrictEnd	RestrictionEnd	DateTimeStamp	0..16,777,216
TicketRestrictLocation	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
TicketRestrictStart	RestrictionStart	DateTimeStamp	0..16,777,216
TicketSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
TicketText	UserData	Databin	40 octets
TicketValidityJourneys	CountOfJourneys	Counter	0..65,535
TicketZoneId	ZoneId	ReferenceIdentifer	2 octets
UnblockInstanceNumber	UnblockInstanceNumber	SequenceNumber	0..65,535
UserData	UserData	Databin	40 octets
UserDataOwner	CompanyId	ReferenceNumber	0..65,535
ValidityAccompaniedBy	AccompaniedBy	SEQUENCE	type EN 1545 number 0..255
ValidityContractDependencyPointer	ContractDependencyPointer	InstancePointer	0..4,294,967,295
ValidityNotVia	LocationIdentifier	SEQUENCE	companyid 0..65,535 LocTypeCode EN 1545 Referenceidentifier 8 octets
ValidityNumberofAdults	NumberOfAdults	Quantity	0..255
ValidityNumberofChildren	NumberOfChildren	Quantity	0..255
ValiditySerialNumber	SerialNumber	ReferenceNumber	0..65,535
ValidityTariffNumber	TariffNumber	ReferenceNumber	0..32,767
ValidityText	UserData	Databin	40 bytes
WrapperApplicationIdentifier		ReferenceIdentifier	8 octets
WrapperSamSerialNumber	SerialNumber	ReferenceNumber	0..4,294,967,295
ZoneListZoneId	ZoneId X n	ReferenceIdentifer	80 octets

C.4 Application data elements not included in EN 1545

It should be noted that an implementer optionally may include any or all of these data elements in their application environment by including them in the user data element within an appropriate data structure. The layout of these elements and whether or not they are in simple data format or expressed as TLV data objects is entirely up to the implementer.

Table C.3 — Application data elements not included in EN 1545

APPLICATION USAGE	DATA TYPE	SPECIFICATION
ApplicationEnvironmentInstanceIdentifierFixed	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
ApplicationEnvironmentInstanceIdentifierVariable	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
ApplicationEnvironmentLabel	SEQUENCE	CompanyId 0..65,535 ExpiryDate DateCompact 0..65,535 ProductId 0..65,535
ApplicationEnvironmentSealFixed	Binary String	16 octets
ApplicationEnvironmentSealVariable	Binary String	16 octets
CtaInstanceIdentifierFixed	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
CtaInstanceIdentifierVariable	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
CtaLabel	SEQUENCE	CompanyId 0..65,535 ExpiryDate DateCompact 0..65,535 ProductId 0..65,535
CtaSealFixed	Binary String	16 octets
CtaSealVariable	Binary String	16 octets
EntitlementInstanceIdentifierFixed	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15

Table C.3 (*continued*)

APPLICATION USAGE	DATA TYPE	SPECIFICATION
EntitlementInstanceIdentifierVariable	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
EntitlementLabel	SEQUENCE	CompanyId 0..65,535 ExpiryDate DateCompact 0..65,535 ProductId 0..65,535
EntitlementSealFixed	Binary String	16 octets
EntitlementSealVariable	Binary String	16 octets
EventLogInstanceIdentifierFixed	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
EventLogInstanceIdentifierVariable	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
EventLogLabel	SEQUENCE	CompanyId 0..65,535 ExpiryDate DateCompact 0..65,535 ProductId 0..65,535
EventLogSealFixed	Binary String	16 octets
EventLogSealVariable	Binary String	16 octets
HolderInstanceIdentifierFixed	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
HolderInstanceIdentifierVariable	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
HolderLabel	SEQUENCE	CompanyId 0..65,535 ExpiryDate DateCompact 0..65,535 ProductId 0..65,535
HolderSealFixed	Binary String	16 octets

Table C.3 (*continued*)

APPLICATION USAGE	DATA TYPE	SPECIFICATION
HolderSealVariable	Binary String	16 octets
StrInstanceIdentifierFixed	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
StrInstanceIdentifierVariable	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
StrLabel	SEQUENCE	CompanyId 0..65,535 ExpiryDate DateCompact 0..65,535 ProductId 0..65,535
StrSealFixed	Binary String	16 octets
StrSealVariable	Binary String	16 octets
TicketInstanceIdentifierFixed	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
TicketInstanceIdentifierVariable	SEQUENCE	unblockInstanceNumber 0..15 productStatus 0..127 versionNumber 0..255 sequencNumber 0..16,777,215 securityVersion 0..65,535 priority 0..15
TicketLabel	SEQUENCE	CompanyId 0..65,535 ExpiryDate DateCompact 0..65,535 ProductId 0..65,535
TicketSealFixed	Binary String	16 octets
TicketSealVariable	Binary String	16 octets

Note that the seal on the Machine Readable Card may be any length according to the sealing method used (CRC or certificate etc.). Typically for space saving reasons, a seal will be 8 bytes or less. In the above Table C.3 16 bytes have been allowed as a maximum value. Where seals are longer than this, they may be truncated or dealt with in full elsewhere.

Annex D
(normative)

ASN.1 Tag allocations

The data as presented at the logical interface within the terminal shall be coded according to ASN.1 BER encoding rules. IMPLICIT Tags shall be used throughout.

It is noted that EN 1545 allocates EXPLICIT Tags to data elements, however since application data structures may contain multiple EN1545 Data Elements of the same type, for example CompanyId used in more than one context within an Application Data Structure, EN 1545 Tags alone will not ensure unambiguous meaning to the Data Elements such that Application IMPLICIT Tags would also have to be present. For this reason EN 1545 EXPLICIT Tags have not been used.

```
INTEROPERABLE PUBLIC TRANSPORT APPLICATION DEFINITIONS AUTOMATIC TAGS ::=
BEGIN

IMPORTS
    AccommodationClassCode, AccompaniedBy, AccountingId, AccountNumber, AssistanceTypeCode,
    Authenticator, Balance, BirthDate, BirthName, BirthPlace, CoachId, CompanyId, CompanyName,
    ContractDependencyPointer, CountOfCoupons, CountOfJourneys, CountryAlpha, CouponsDeducted,
    CouponsLoaded, CumulativeFare, Date, DateTimeStamp, Destination, DeviceId, DeviceTypeCode,
    DirectionCode, DirectionCode, DiscountCode, DossierId, EmailAddress, EndDate, EndTime,
    EndTimeStamp, EntitlementTypeCode, EventTypeCode, ExtraServiceCode, FareDeducted, Fax,
    Forename, GenderCode, HolderAddress, HolderId, HotListStatusCode, IdentityDocumentId,
    InterchangesAllowed, JourneyDistance, JourneyRunId, LanguageId, LastMinuteSale,
    LegislationCode, LocationIdentifier, LockTime, LoyaltyPoints, LoyaltySchemeId,
     mappingType,MaxAmountLimit, MembershipId, MinAmountLimit, MostRecentPointer, NotOKCounter,
     NumberOfAdults, NumberOfChildren, NumberOfPassbacks, Origin, OverbookingIndicator,
     PassbackTime, PassengerTotal, PaymentMeansCode, PaymentUnit, PeriodJourneys, PostcodeId, Price,
     Priority, ProductId, ProductStatus, ProfileCodeIOP, ReceiptData, ReceiptPrintedFlag,
     ReferenceNumber, ReservationReferenceId, RestrictedDayOfWeek, RestrictedPeriodOfDay,
     Restriction, RestrictionEnd, RestrictionStart, ResultCode, RouteId, RouteVariantId,
     SeatAlphaId, SeatNumber, SeatPositionCode, SecondaryFlag, SecurityVersion, SequenceNumber,
     SerialNumber, SeriousnessCode, Smoking, StartDate, StartTime, StrLoadCode, Surname,
     TariffNumber, Telephone, ThresholdAmount, Time, TransactionSequenceNumber, TransportTypeCode,
     TravelServiceId, UnblockInstanceNumber, UserActionCode, UserData, ValidationCounter,
     ValidationModelCode,ValidityDuration, VehicleId, VersionNumber, ZoneId, ZoneMap
FROM TransportGeneral2
    { iso (1) identifiedorg(3) cen.std (?) EN1545 (1545) abstractsyntax (1)
    transportGeneral2 (2)
    }

EXPORTS ALL;

[APPLICATION 00] M-ApplicationEnvironment ::=     SEQUENCE
{   applicationEnvironmentLabel                       M-ProductLabel
,   applicationEnvironmentInstanceIdentifierFixed     M-ProductInstanceIdentifier
,   applicationEnvironmentSealFixed                   M-Seal
,   applicationEnvironmentInstanceIdentifierVariable  M-ProductInstanceIdentifier   OPTIONAL
,   applicationEnvironmentSealVariable                M-Seal                         OPTIONAL
,   applicationEnvironmentSamSerialNumber             SerialNumber
,   applicationEnvironmentDefaultCurrency             PaymentUnit
,   applicationEnvironmentMappingType                 MappingType
,   applicationEnvironmentSaleDate                    StartDate
,   applicationEnvironmentMasterTransactionCounter    TransactionSequenceNumber
,   validityPeriod                                    A-Validity Period              OPTIONAL
,   deposit                                           A-Deposit                      OPTIONAL
,   blockingStatus                                    A-BlockingStatus               OPTIONAL
,   event                                             SEQUENCE OF CHOICE
    {   primarySavedEventData                             L-PrimarySavedEventData
    ,   secondarySavedEventData                           L-SecondarySavedEventData
    ,   problemLog                                        L-ProblemLog
    }                                                                                OPTIONAL
,   userSpecific                                      L-UserSpecific                 OPTIONAL
}

[APPLICATION 01] M-Holder ::=                     SEQUENCE
{   holderLabel                                       M-ProductLabel
,   holderInstanceIdentifierFixed                     M-ProductInstanceIdentifier
```

```
,   holderSealFixed                                 M-Seal
,   holderInstanceIdentifierVariable                M-ProductInstanceIdentifier   OPTIONAL
,   holderSealVariable                              M-Seal                        OPTIONAL
,   holderSamSerialNumber                           SerialNumber
,   validityPeriod                                  A-Validity Period             OPTIONAL
,   holderPersonalDetail                            A-HolderPersonalDetail        OPTIONAL
,   holderCorporateDetail                           A-HolderCorporateDetail       OPTIONAL
,   customerPreferences                             A-CustomerPreferences         OPTIONAL
,   customerEntitlementDetail                       A-CustomerEntitlementDetail   OPTIONAL
,   ticketRestrictions                              A-TicketRestrictions          OPTIONAL
,   blockingStatus                                  A-BlockingStatus              OPTIONAL
,   event                                           SEQUENCE OF CHOICE
    {   primarySavedEventData                           L-PrimarySavedEventData
    ,   secondarySavedEventData                         L-SecondarySavedEventData
    }                                                                             OPTIONAL
,   userSpecific                                    L-UserSpecific                OPTIONAL
}

[APPLICATION 02] M-EventLog ::=                     SEQUENCE
{   eventLogLabel                                       M-EventLogLabel
,   eventLogInstanceIdentifierFixed                     M-EventLogInstanceIdentifier
,   eventLogSealFixed                                   M-Seal
,   eventLogInstanceIdentifierVariable                  M-EventLogInstanceIdentifier  OPTIONAL
,   eventLogSealVariable                                M-Seal                        OPTIONAL
,   eventLogSamSerialNumber                             SerialNumber
,   eventLog                                        SEQUENCE OF L-EventLog
}

[APPLICATION 03] M-ChargeToAccount ::=              SEQUENCE
{   ctaLabel                                            M-ProductLabel
,   ctaInstanceIdentifierFixed                          M-ProductInstanceIdentifier
,   ctaSealFixed                                        M-Seal
,   ctaInstanceIdentifierVariable                       M-ProductInstanceIdentifier   OPTIONAL
,   ctaSealVariable                                     M-Seal                        OPTIONAL
,   ctaSamSerialNumber                                  SerialNumber
,   productData                                         M-ProductData ( WITH COMPONENTS
    {   validityPeriod                                      OPTIONAL
    ,   retailer                                            OPTIONAL
    ,   deposit                                             ABSENT
    ,   loyalty                                             OPTIONAL
    ,   ctaAdditionalInfo                                   OPTIONAL
    ,   storedTravelRightsAutoLoad                          ABSENT
    ,   rideBasedTravelLoad                                 ABSENT
    ,   storedTravelRightsInformation                       ABSENT
    ,   customerEntitlementDetail                           ABSENT
    ,   ticketGeography                                     ABSENT
    ,   ticketValidityInformation                           ABSENT
    ,   ticketRestrictions                                  ABSENT
    ,   ticketServiceValidity                               ABSENT
    ,   reservationBasic                                    ABSENT
    ,   reservationAdditional                               ABSENT
    ,   ticketSpecialConditions                             ABSENT
    ,   blockingStatus                                      OPTIONAL
    ,   storedTravelRightsBalance                           ABSENT
    ,   storedTravelRightsLoad                              ABSENT
    ,   strCtaUsage                                         OPTIONAL
    ,   rideBasedTravelUsage                                ABSENT
    ,   rideBasedTravelLoad                                 ABSENT
    ,   loyaltyBalance                                      OPTIONAL
    ,   event                                               OPTIONAL
    ,   zoneList                                            OPTIONAL
    ,   userSpecific                                        OPTIONAL
    ,   paymentReceipt                                      OPTIONAL
    } )
}


[APPLICATION 04] M-StoredTravelRights ::=           SEQUENCE
{   strLabel                                            M-ProductLabel
,   strInstanceIdentifierFixed                          M-ProductInstanceIdentifier
,   strSealFixed                                        M-Seal
,   strInstanceIdentifierVariable                       M-ProductInstanceIdentifier   OPTIONAL
,   strSealVariable                                     M-Seal                        OPTIONAL
,   strSamSerialNumber                                  SerialNumber
,   strUnit                                             PaymentUnit
,   strMaxValue                                         MaxAmountLimit
```

```
,   productData                                 M-ProductData ( WITH COMPONENTS
    {   validityPeriod                              OPTIONAL
    ,   retailer                                    OPTIONAL
    ,   deposit                                     OPTIONAL
    ,   loyalty                                     OPTIONAL
    ,   ctaAdditionalInfo                           ABSENT
    ,   storedTravelRightsAutoLoad                  OPTIONAL
    ,   rideBasedTravelLoad                         ABSENT
    ,   storedTravelRightsInformation               OPTIONAL
    ,   customerEntitlementDetail                   ABSENT
    ,   ticketGeography                             ABSENT
    ,   ticketValidityInformation                   ABSENT
    ,   ticketRestrictions                          ABSENT
    ,   ticketServiceValidity                       ABSENT
    ,   reservationBasic                            ABSENT
    ,   reservationAdditional                       ABSENT
    ,   ticketSpecialConditions                     ABSENT
    ,   blockingStatus                              OPTIONAL
    ,   storedTravelRightsBalance                   OPTIONAL
    ,   storedTravelRightsLoad                      OPTIONAL
    ,   strCtaUsage                                 OPTIONAL
    ,   rideBasedTravelUsage                        ABSENT
    ,   rideBasedTravelLoad                         ABSENT
    ,   loyaltyBalance                              OPTIONAL
    ,   event                                       OPTIONAL
    ,   zoneList                                    OPTIONAL
    ,   userSpecific                                OPTIONAL
    ,   paymentReceipt                              OPTIONAL
    } )
}

[APPLICATION 05] M-CustomerEntitlement ::=      SEQUENCE
{   entitlementLabel                            M-ProductLabel
,   entitlementInstanceIdentifierFixed          M-ProductInstanceIdentifier
,   entitlementSealFixed                        M-Seal
,   entitlementInstanceIdentifierVariable       M-ProductInstanceIdentifier   OPTIONAL
,   entitlementSealVariable                     M-Seal                        OPTIONAL
,   entitlementSamSerialNumber                  SerialNumber
,   entitlementType                             EntitlementTypeCode
,   entitlementDossierReference                 DossierId
,   productData                                 M-ProductData ( WITH COMPONENTS
    {   validityPeriod                              OPTIONAL
    ,   retailer                                    OPTIONAL
    ,   deposit                                     ABSENT
    ,   loyalty                                     OPTIONAL
    ,   ctaAdditionalInfo                           ABSENT
    ,   storedTravelRightsAutoLoad                  ABSENT
    ,   rideBasedTravelLoad                         ABSENT
    ,   storedTravelRightsInformation               ABSENT
    ,   customerEntitlementDetail                   OPTIONAL
    ,   ticketGeography                             ABSENT
    ,   ticketValidityInformation                   OPTIONAL
    ,   ticketRestrictions                          OPTIONAL
    ,   ticketServiceValidity                       OPTIONAL
    ,   reservationBasic                            ABSENT
    ,   reservationAdditional                       ABSENT
    ,   ticketSpecialConditions                     ABSENT
    ,   blockingStatus                              OPTIONAL
    ,   storedTravelRightsBalance                   ABSENT
    ,   storedTravelRightsLoad                      ABSENT
    ,   strCtaUsage                                 ABSENT
    ,   rideBasedTravelUsage                        ABSENT
    ,   rideBasedTravelLoad                         ABSENT
    ,   loyaltyBalance                              OPTIONAL
    ,   event                                       OPTIONAL
    ,   zoneList                                    OPTIONAL
    ,   userSpecific                                OPTIONAL
    ,   paymentReceipt                              ABSENT
    } )
}

[APPLICATION 06] M-Ticket ::=                   SEQUENCE
{   ticketLabel                                 M-ProductLabel
,   ticketInstanceIdentifierFixed               M-ProductInstanceIdentifier
,   ticketSealFixed                             M-Seal
,   ticketInstanceIdentifierVariable            M-ProductInstanceIdentifier   OPTIONAL
```

```
,   ticketSealVariable                          M-Seal   OPTIONAL
,   ticketSamSerialNumber                       SerialNumber
,   ticketPrice                                 Price
,   productData                                 M-ProductData ( WITH COMPONENTS
    ,   validityPeriod                              OPTIONAL
    ,   retailer                                    OPTIONAL
    ,   deposit                                     ABSENT
    ,   loyalty                                     OPTIONAL
    ,   ctaAdditionalInfo                           ABSENT
    ,   storedTravelRightsAutoLoad                  ABSENT
    ,   rideBasedTravelLoad                         OPTIONAL
    ,   storedTravelRightsInformation               ABSENT
    ,   customerEntitlementDetail                   ABSENT
    ,   ticketGeography                             OPTIONAL
    ,   ticketValidityInformation                   OPTIONAL
    ,   ticketRestrictions                          OPTIONAL
    ,   ticketServiceValidity                       OPTIONAL
    ,   reservationBasic                            OPTIONAL
    ,   reservationAdditional                       OPTIONAL
    ,   ticketSpecialConditions                     OPTIONAL
    ,   blockingStatus                              OPTIONAL
    ,   storedTravelRightsBalance                   ABSENT
    ,   storedTravelRightsLoad                      ABSENT
    ,   strCtaUsage                                 ABSENT
    ,   rideBasedTravelUsage                        OPTIONAL
    ,   rideBasedTravelLoad                         OPTIONAL
    ,   loyaltyBalance                              OPTIONAL
    ,   event                                       OPTIONAL
    ,   zoneList                                    OPTIONAL
    ,   userSpecific                                OPTIONAL
    ,   paymentReceipt                              ABSENT
    } )
}

[APPLICATION 07] M-Wrapper ::=              SEQUENCE
{   wrapperLabel                                M-ProductLabel
,   wrapperInstanceIdentifierFixed              M-ProductInstanceIdentifier
,   wrapperSealFixed                            M-Seal
,   wrapperInstanceIdentifierVariable           M-ProductInstanceIdentifier   OPTIONAL
,   wrapperSealVariable                         M-Seal   OPTIONAL
,   wrapperSamSerialNumber                      SerialNumber
,   wrapperApplicationIdentifier                ReferenceNumber
,   wrapperData                                 OCTET STRING
}



M-ProductData ::=                           SEQUENCE
{   validityPeriod                              A-Validity Period               OPTIONAL
,   retailer                                    A-Retailer                      OPTIONAL
,   deposit                                     A-Deposit                       OPTIONAL
,   loyalty                                      A-Loyalty                       OPTIONAL
,   ctaAdditionalInfo                           A-CTA-AdditionalInfo            OPTIONAL
,   storedTravelRightsAutoLoad                  A-StoredTravelRightsAutoLoad    OPTIONAL
,   rideBasedTravelLoad                         A-RideBasedTravelLoad           OPTIONAL
,   storedTravelRightsInformation               A-StoredTravelRightsInformation OPTIONAL
,   customerEntitlementDetail                   A-CustomerEntitlementDetail     OPTIONAL
,   ticketGeography                             A-TicketGeography               OPTIONAL
,   ticketValidityInformation                   A-TicketValidityInformation     OPTIONAL
,   ticketRestrictions                          A-TicketRestrictions            OPTIONAL
,   ticketServiceValidity                       A-TicketServiceValidity         OPTIONAL
,   reservationBasic                            A-ReservationBasic              OPTIONAL
,   reservationAdditional                       A-ReservationAdditional         OPTIONAL
,   ticketSpecialConditions                     A-TicketSpecialConditions       OPTIONAL
,   blockingStatus                              A-BlockingStatus                OPTIONAL
,   storedTravelRightsBalance                   L-StoredTravelRightsBalance     OPTIONAL
,   storedTravelRightsLoad                      L-StoredTravelRightsLoad        OPTIONAL
,   strCtaUsage                                 L-StrCtaUsage                   OPTIONAL
,   rideBasedTravelUsage                        L-RideBasedTravelUsage          OPTIONAL
,   rideBasedTravelLoad                         L-RideBasedTravelLoad           OPTIONAL
,   loyaltyBalance                              L-LoyaltyBalance                OPTIONAL
,   event                                       SEQUENCE OF CHOICE
    {   primarySavedEventData                       L-PrimarySavedEventData
    ,   secondarySavedEventData                     L-SecondarySavedEventData
    }                                                                           OPTIONAL
,   zoneList                                    L-ZoneList                      OPTIONAL
```

```
,   userSpecific                                    L-UserSpecific                  OPTIONAL
,   paymentReceipt                                  L-PaymentReceipt                OPTIONAL
,   }

--  Mandatory Data Structures - Type M

M-ApplicationEnvironmentLabel ::=               SEQUENCE
{   applicationOwner                                CompanyId
,   expiryDate                                      EndDate
}

M-ApplicationEnvironmentInstanceIdentifier ::=  SEQUENCE
{   unblockInstanceNumber                           UnblockInstanceNumber
,   status                                          ProductStatus
,   versionNumber                                   VersionNumber
,   dataGroupSequenceNumber                         SequenceNumber
,   security                                        SecurityVersion
}

M-Seal                                          Authenticator

M-ProductLabel ::=                              SEQUENCE
{   productOwner                                    CompanyId
,   expiryDate                                      EndDate
,   productId                                       ProductId
}

M-ProductInstanceIdentifier ::=                 SEQUENCE
{   unblockInstanceNumber                           UnblockInstanceNumber
,   status                                          ProductStatus
,   versionNumber                                   VersionNumber
,   dataGroupSequenceNumber                         SequenceNumber
,   security                                        SecurityVersion
,   priority                                        Priority
}

M-EventLogLabel ::=                             SEQUENCE
{   applicationOwner                                CompanyId
,   expiryDate                                      EndDate
}

M-EventLogInstanceIdentifier ::=                SEQUENCE
{   versionNumber                                   VersionNumber
,   dataGroupSequenceNumber                         SequenceNumber
,   security                                        SecurityVersion
,   mostRecentPointer                               MostRecentPointer
}

--  Optional Data Structures - Type A

A-ValidityPeriod ::=                            SEQUENCE
{   validityStartDate                               StartDate
,   validityEndDate                                 EndDate
,   validityStartTime                               StartTime
,   validityEndTime                                 EndTime
}

A-Retailer ::=                                  SEQUENCE
{   retailerIdentifier                              CompanyId
,   retailerSaleAgent                               CompanyId
,   retailerSaleDevice                              DeviceId
}

A-Deposit ::=                                   Price

A-Loyalty ::=                                   SEQUENCE
{   loyaltyMembershipID                             MembershipId
,   loyaltySchemeId                                 LoyaltySchemeId
}

A-HolderPersonalDetail ::=                      SEQUENCE
{   holderSecondaryFlag                             SecondaryFlag
,   holderSurname                                   Surname
,   holderForename                                  Forename
,   holderGender                                    GenderCode
,   holderBirthName                                 BirthName
```

```
,   holderBirthDate                         BirthDate
,   holderBirthPlace                        BirthPlace
,   holderProfile                           ProfileCodeIOP
,   holderIdNumber                          HolderId
,   holderCountryAlpha                      CountryAlpha
,   holderAddress                           HolderAddress
,   holderZipcode                           PostCodeId
,   holderTelephone                         Telephone
,   holderMobile                            Telephone
,   holderFax                               Fax
,   holderEmail                             EmailAddress
,   holderLanguage                          LanguageId
,   holderIdDocumentNumber                  IdentityDocumentId
}

A-HolderCorporateDetail ::=                 SEQUENCE
{   holderCompanyName                           CompanyName
,   holderCompanyAddress                        HolderAddress
,   holderCompanyZipcode                        PostcodeId
,   holderCompanyTelephone                      Telephone
}

A-CustomerPreferences ::=                   SEQUENCE
{   preferencePassengerClass                    AccommodationClassCode
,   preferenceSmoker                            Smoking
,   preferencePaymentMeans                      PaymentMeansCode
,   preferenceDirection                         DirectionCode
}

A-CtaAdditionalInfo ::=                     SEQUENCE
{   ctaTransactionLimit                         Price
,   ctaAccountingReference                      AccountingId
}

A-StoredTravelRightsAutoLoad ::=            SEQUENCE
{   strAutoLoadThreshold                        ThresholdAmount
,   strAutoLoadValue                            MaxAmountLimit
}

A-RideBasedTravelLoad ::=                   SEQUENCE
{   rbtMaxLoadLimit                             MaxAmountLimit
,    rbtMaxLoadValue                             MaxAmountLimit
,   rbtValue                                    MaxAmountLimit
,   rbtUnitType                                 TravelServiceId
,   rbtReuseLockTime                            LockTime
}

A-StoredTravelRightsInformation ::=         SEQUENCE
{   strSerialNumber                             SerialNumber
,   strMaxTransactionValue                      MaxAmountLimit
,   strMinTransactionValue                      MinAmountLimit
,   strReloadSelection                          StrLoadCode
}

A-CustomerEntitlementDetail ::=             SEQUENCE
{   entitlementClass                            AccommodationClassCode
,   entitlementType                             EntitlementTypeCode
,   entitlementPassengerTotal                   PassengerTotal
,   entitlementSerialNumber                     SerialNumber
,   entitlementPrice                            Price
,   entitlementReference                        AccountingId
,   entitlementDiscount                         DiscountCode
}

A-TicketGeography ::=                       SEQUENCE
{   ticketJourneyOrigin                         Origin
,   ticketJourneyDestination                    Destination
,   ticketZoneMap                               ZoneMap
,   ticketZoneId                                ZoneId
,   ticketText                                  UserData
}

A-TicketValidityInformation ::=             SEQUENCE
{   validityRemoveDate                          Date
,   validitySaleDateTimeStamp                   DateTimeStamp
,   validityDepartureDateTimeStamp              DateTimeStamp
```

```
,   validityPassengerClass                  AccommodationClassCode
,   validityAccompaniedBy                   AccompaniedBy
,   validityNumberofAdults                  NumberOfAdults
,   validityNumberofChildren                NumberOfChildren
,   validityPaymentMeans                    PaymentMeansCode
,   validityTariffNumber                    TariffNumber
,   validitySerialNumber                    SerialNumber
,   validityTrainCategory                   TransportTypeCode
,   validityNotVia                          LocationIdentifier
,   validityContractDependencyPointer       ContractDependencyPointer
,   validityValidationModel                 ValidationModelCode
,   validityText                            UserData
}

A-TicketRestrictions                        SEQUENCE
{   ticketJourneyInterchanges               InterchangesAllowed
,   ticketJourneyDistance                   JourneyDistance
,   ticketJourneyOrigin                     LocationIdentifier
,   ticketJourneyDestination                LocationIdentifier
,   ticketJourneyRun                        JourneyRunId
,   ticketJourneyVia                        LocationIdentifier
,   ticketPeriodJourneys                    PeriodJourneys
,   ticketRestriction                       Restriction
,   ticketRestrictEnd                       RestrictionEnd
,   ticketRestrictLocation                  LocationIdentifier
,   ticketRestrictStart                     RestrictionStart
,   ticketRestrictedPeriodOfDay             RestrictedPeriodOfDay
,   ticketServices                          TransportTypeCode
,   ticketValidityJourneys                  CountOfJourneys
,   ticketValidityLimitDate                 Date
,   ticketVehicleClassAllowed               AccommodationClassCode
,   ticketFurthestPlace                     LocationIdentifier
,   ticketEndTimeStamp                      EndTimeStamp
,   ticketNumberOfPassbacks                 NumberOfPassbacks
,   ticketPassbackTime                      PassbackTime
,   ticketRestrictDOW                       RestrictedDayOfWeek
,   ticketZoneId                            ZoneId
}

A-TicketServiceValidity ::=                 SEQUENCE
{   ticketJourneyRoute                      RouteId
,   ticketJourneyRouteVariant               RouteVariantId
}

A-ReservationBasic ::=                      SEQUENCE
{   reservationServiceId                    TravelServiceId
,   reservationDepartureDateTimeStamp       DateTimeStamp
,   reservationCoachIdFrom                  CoachId
,   reservationCoachIdTo                    CoachId
,   reservationCoachAlphaFrom               SeatAlphaId
,   reservationCoachAlphaTo                 SeatAlphaId
,   reservationSeatNumberFrom               SeatNumber
,   reservationSeatNumberTo                 SeatNumber
,   reservationSeatAlphaFrom                SeatAlphaId
,   reservationSeatAlphaTo                  SeatAlphaId
,   reservationPlaceType                    SeatPositionCode
}

A-ReservationAdditional ::=                 SEQUENCE
{   reservationNumber                       ReservationReferenceId
,   reservationOverbookingIndicator         OverbookingIndicator
,   reservationPhysicalClass                AccommodationClassCode
,   reservationServiceOperator              CompanyId
,   reservationSmokers                      Smoking
,   reservationSpecialFacility              AssistanceTypeCode
,   reservationArrivalDateTimeStamp         DateTimeStamp
,   reservationDossierReference             DossierId
,   reservationService                      ExtraServiceCode
,   reservationOrigin                       LocationIdentifier
,   reservationDestination                  LocationIdentifier
}

A-TicketSpecialConditions ::=               SEQUENCE
{   ticketLastMinuteSale                    LastMinuteSale
,   ticketLegislativeBackground             LegislationCode
}
```

```
A-BlockingStatus ::=                    SEQUENCE
{   blockingDateTimeStamp                   DateTimeStamp
,   blockingReason                          HotListStatusCode
,   blockingSamSerialNumber                 SerialNumber
}

--  Optional Data Structures - Type L

L-ProblemLog ::=                        SEQUENCE
{   problemDateTimeStamp                    DateTimeStamp
,   problemEventCode                        EventTypeCode
,   problemSamSerialNumber                  SerialNumber
,   problemDevice                           DeviceId
,   problemDeviceSerialNumber               SerialNumber
,   problemResult                           ResultCode
,   problemNotOKCounter                     NotOKCounter
,   problemSeriousness                      SeriousnessCode
}

L-StoredTravelRightsBalance ::=         Balance

L-StoredTravelRightsLoad ::=            SEQUENCE
{   strDateTimeStamp                        DateTimeStamp
,   strSequenceNumber                       TransactionSequenceNumber
,   strSamSerialNumber                      SerialNumber
,   strRetailerId                           CompanyId
,   strAmountLoaded                         MaxAmountLimit
,   strAfterLoadBalance                     Balance
,   strReceiptFlag                          ReceiptPrintedFlag
}

L-StrCtaUsage ::=                       SEQUENCE
{   strCtaFareDeducted                      FareDeducted
,   strCtaPriceReduction                    DiscountCode
,   strCtaCumulativeFare                    CumulativeFare
,   strCtaDateOfLastRide                    Date
,   strCtaTimeOfLastRide                    Time
,   strCtaCountOfJourneys                   CountOfJourneys
,   strCtaServiceOfLastRide                 TravelServiceId
,   strCtaLocationIdentifier                LocationIdentifier
,   strCtaContractDependencyPointer         SequenceNumber
}

L-RideBasedTravelUsage ::=              SEQUENCE
{   rbtCouponsDeducted                      CouponsDeducted
,   rbtCountOfCoupons                       CountOfCoupons
}

L-RideBasedTravelLoad ::=               SEQUENCE
{   rbtDateTimeStamp                        DateTimeStamp
,   rbtSamSerialNumber                      SerialNumber
,   rbtRetailerIdentifier                   CompanyId
,   rbtCouponsLoaded                        CouponsLoaded
,   rbtLoadPrice                            Price
,   rbtAfterLoadBalance                     Balance
}

L-LoyaltyBalance ::=                    SEQUENCE
{   loyaltyPoints                           LoyaltyPoints
,   loyaltyValidityDuration                 ValidityDuration
,   loyaltySchemeId                         LoyaltySchemeId
}

L-PrimarySavedEventData ::=             SEQUENCE
{   savedEventTypeCode                      EventTypeCode
,   savedEventSequenceNumber                TransactionSequenceNumber
,   savedEventDateTimeStamp                 DateTimeStamp
,   savedEventResultCode                    ResultCode
,   savedEventDeviceSerialNumber            SerialNumber
,   savedEventDeviceTypeCode                DeviceTypeCode
,   savedEventServiceOperator               CompanyId
,   savedEventJourneyOrigin                 LocationIdentifier
,   savedEventJourneyDestination            LocationIdentifier
}
```

```
L-SecondarySavedEventData ::=                SEQUENCE
{   savedEventStatusReceiptFlag                 ReceiptPrintedFlag
,   savedEventDeviceOwner                       CompanyId
,   savedEventDevice                            DeviceId
,   savedEventDeviceSequenceNumber              SequenceNumber
,   savedEventJourneyVia                        LocationIdentifier
,   savedEventJourneyRoute                      RouteId
,   savedEventJourneyRouteVariant               RouteVariantId
,   savedEventJourneyRun                        JourneyRunId
,   savedEventDirection                         DirectionCode
,   savedEventCountOfJourneys                   CountOfJourneys
,   savedEventTotalJourneys                     CountOfJourneys
,   savedEventTicketUseValidityDate             EndDate
,   savedEventTicketUseValidityTime             EndTime
,   savedEventJourneyDistance                   JourneyDistance
,   savedEventJourneyInterchanges               InterchangesAllowed
,   savedEventZoneId                            ZoneId
,   savedEventPricePaid                         Price
,   savedEventPaymentMeans                      PaymentMeansCode
,   savedEventNonDiscountedPrice                Price
,   savedEventValidationCounter                 ValidationCounter
,   savedEventValidationStatus                  ProductStatus
,   savedEventTransportServiceCode              TransportTypeCode
,   savedEventVehicleClass                      AccommodationClassCode
,   savedEventVehicleId                         VehicleId
,   savedEventLocationIdentifier                LocationIdentifier
,   savedEventPassengerClass                    AccommodationClassCode
,   savedEventCustomerProfile                   ProfileCodeIOP
,   savedEventPassengerTotal                    PassengerTotal
,   savedEventText                              UserData
}
L-ZoneList ::=                               SEQUENCE OF ZoneId

L-UserSpecific ::=                           SEQUENCE
{   userDataOwner                               CompanyId
,   userData                                    UserData
}

L-PaymentReceipt ::=                         SEQUENCE
{   paymentCertificateAccountNumber             AccountNumber
,   paymentCertificatePaymentMeans              PaymentMeansCode
,   paymentCertificatePaymentProvider           CompanyId
,   paymentCertificateData                      UserData
,   paymentCertificatePrice                     Price
,   paymentCertificateStrBalance                Balance
,   paymentReceiptAccountNumber                 AccountNumber
,   paymentReceiptData                          ReceiptData
,   paymentReceiptDate                          Date
,   paymentReceiptDevice                        DeviceId
,   paymentReceiptLocationIdentifier            LocationIdentifier
,   paymentReceiptPaymentMeans                  PaymentMeansCode
,   paymentReceiptPaymentProvider               CompanyId
,   paymentReceiptPrice                         Price
,   paymentReceiptStrBalance                    Balance
,   paymentReceiptResult                        ResultCode
,   paymentReceiptSerialNumber                  SerialNumber
,   paymentReceiptServiceOperator               CompanyId
,   paymentPayServicePoint                      TransportTypeCode
,   paymentPayServicePointInfo                  UserActionCode
}


L-EventLog ::=                                 SEQUENCE
{   eventTypeCode                               EventTypeCode
,   eventSequenceNumber                         TransactionSequenceNumber
,   eventContractDependencyPointer              SequenceNumber
,   eventDateTimestamp                          DateTimeStamp
,   eventPassbackTime                           PassbackTime
,   eventResultCode                             ResultCode
,   eventSamSerialNumber                        SerialNumber
,   eventDeviceId                               DeviceId
}
END
```

Annex E
(informative)

General requirements

E.1 Introduction

The purpose of this European Standard is to achieve interoperability throughout transport ticketing applications, while making sure that public transport companies remain as commercially free as possible to design their own implementation in pursuing their own business strategies. The standard defines the codification of products in a standardised way. However, this European Standard is not a prescriptive specification, rather it is a foundation for interoperability such that interoperability may be achieved across different implementations of the Interoperable Surface Transport Application subject to commercial agreements and exchange of details of the physical interpretation of the standard by each implementer.

Specific transport requirements include the following:

— user needs to be able to travel with all participating operators (the seamless journey) using a single card;

— there needs to be a capability to extract data appropriate to the revenue-sharing and statistical requirements of the transport operators;

— opportunity may be provided to use the same card for other applications and combine those with the transport application;

— ticketing methods and verification procedures associated with the application offer the opportunity to reduce the current time taken to enter/exit the public transport system and may reduce payment handling costs significantly;

— compliance with European data protection and financial services laws/regulations (e.g. privacy);

— provides a tangible object for the life of the product which is universally legible and identifiable;

— provides an open-ended capability to accommodate new products as required regardless of products already in existence.

E.2 Overview

The transport application life cycle from the point of view of the user considers all the interactions with the card from the initial creation of the application on the card through to its removal. Between these points a number of other activities may occur to add, modify or delete specific transport entitlements. Figures E.1, E.2 and E.3 illustrate the key activities and relationships:

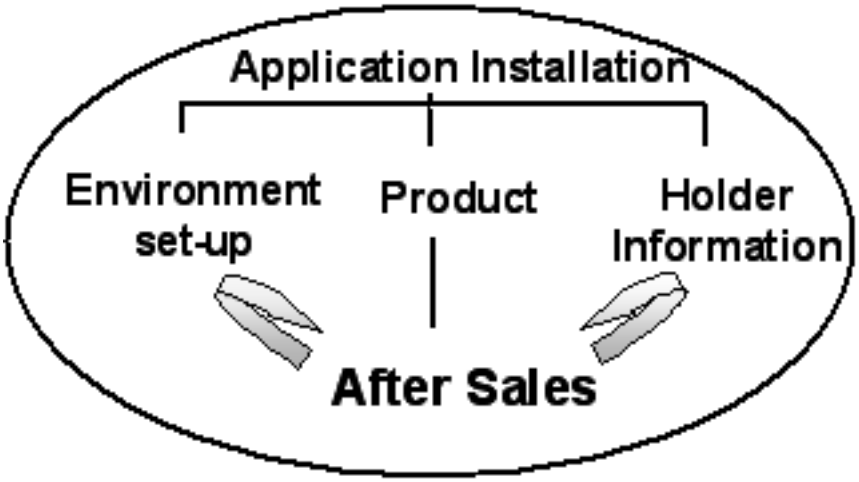

Figure E.1 — The Interopeable Public Transport Application

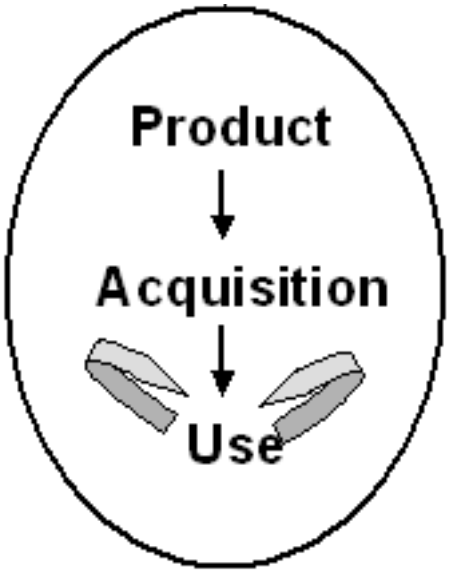

Figure E.2 — Products within the Interoperable Public Transport Application

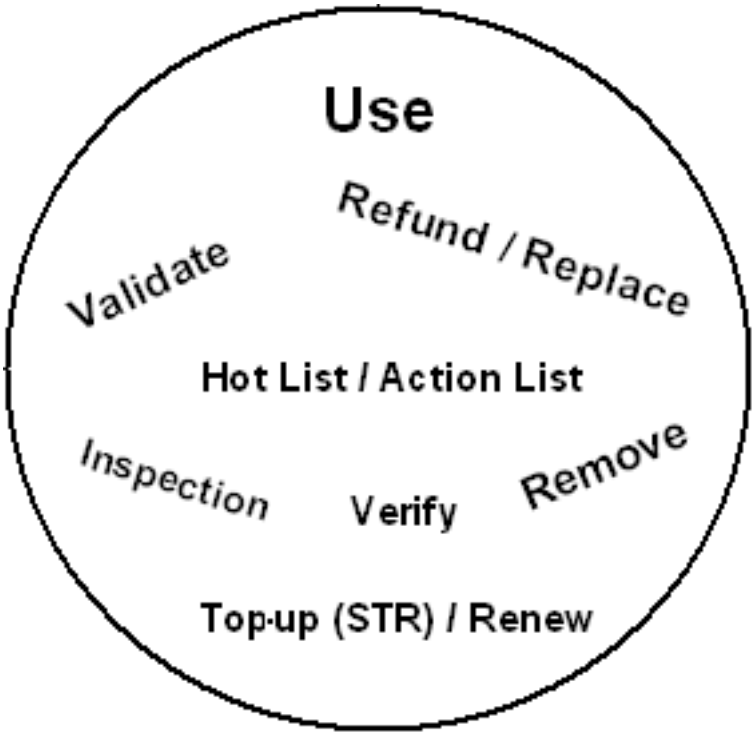

Figure E.3 — Interoperable Public Transport Application product usage

The three figures above may be clarified thus:

Application Installation:

The public transport application is stored on the card, incorporating the scheme identity and security aspects. When the application is set up, it is installed by an accredited application retailer.

Environment Set-up:

Writes information applicable to the operation of this instance of the application.

Holder Information:

If present, writes information pertinent to the holder.

Product Acquisition:

The product retailer writes acquired products to the application

Product Use:

Involves the following potential processes:

— validation;

— verification;

— inspection;

— top-up;

— renewal;

— refund;

— replace;

— hot list;

— remove.

Product Removal (unloading):

May take place any time after usage or expiry date (with any hangover/retention period)

E.3 Entities, actors and functions

These are fully defined as part of an IFM environment in the relevant CEN standard documentation.

E.4 Distribution of products

E.4.1 General

The Interopeable Public Transport Application is a flexibly designed toolkit to meet European ticketing requirements. One difficulty is the different terminology used for ticketing systems across Europe. However, it caters for conventional pre-paid and pay at point-of-use ticketing as well as new approaches related to the payment world such as stored value and Charge to Account.

Examples of these ticketing products are described in the following subclauses:

E.4.2 Conventional prepaid and season tickets

With reference to selling fixed price tickets (application products) prior to a journey (e.g. single ride tickets, season tickets); it shall be possible for a customer to carry out the following activities:

— to obtain a prepaid ticket prior to a number of journeys and immediately before/at the beginning of a specific journey;

— to buy a certain number of prepaid tickets at the same time for more than one journey (e.g. carnets, Streifenkarten).

E.4.3 Post-ride in an Interoperable Fare Management (IFM) system

This refers to the process of enabling a customer to participate in a post-ride Interoperable Fare Management (IFM) system. The application provides the product definitions that may be used within this environment (e.g. BIBO, CICO). This may comprise the proof of a customer's identification and other fields defined by the scheme.

E.4.4 Stored Travel Rights in an Interoperable Fare Management (IFM) system

This comprises the process of making use of a debit facility to load a certain value in a product and if necessary and authorised, to remove any remaining value. It can be used in a BIBO or CICO environment as well as in a pay-on-entry system. This may comprise the proof of a customer's identification and other fields defined by the scheme.

E.5 Commercial requirements for interoperability

E.5.1 General

Interoperability needs to be considered at both a technical and commercial level. This European Standard describes the minimum requirements for interoperability at the logical level. The standard provides a toolkit for achieving these requirements but is transparent as regards the commercial relationships which need to be in place to deliver working systems.

Customers of public transport should be able to use different electronic ticketing systems with the same public transport application on their cards. This means, customers will be able to obtain rights to travel using this public transport application from any participating service operator, regardless of the kind of the electronic ticketing system in place.

E.5.2 Distribution requirements

E.5.2.1 Distribution of cards and acquisition of the public transport application

It needs to be possible to install the interoperable public transport application as specified in this European Standard in a variety of card environments, some examples of which are:

— multi-issuer / multi-application card;

— single-issuer / multi-application card (e.g. city or university card);

— single-issuer / single-application card (e.g. public transport card).

Within these environments, a customer's need for data privacy needs to be ensured. This includes the use of "anonymous cards" which do not contain data related to a specific card holder and personalised cards where there is a requirement to protect personal information which may be held on the card.

E.5.2.2 Acquisition of products

E.5.2.2.1 Customer entitlement

The product retailer issues the customer entitlement based upon the product owner’s requirement.

E.5.2.2.2 Charge To Account (CTA)

The product retailer provides an entitlement to travel as a result of a commercial contract arranged between the holder and the product owner.

E.5.2.2.3 Stored Travel Rights (STR)

This is a form of CTA where the payment is made in advance. Acquisition follows the same process as for CTA.

E.5.2.2.4 Tickets

Tickets are issued by a product retailer and configured according to the commercial rules pertaining. Sales and usage transactions are produced in the same way as for CTA/STR however, in the special case of tickets which are pre-issued but not yet valid for travel, the first usage transaction also generates supplementary product information (e.g. expiry date, which may be dependent upon other card attributes).

E.5.3 Product requirements

Having defined the building blocks above, the following are now possible:

Products:

a) single ride ticket;

b) period pass;

c) special fares (e.g. disabled, pensioner, children);

d) group tickets;

e) stored travel rights;

f) charge to account;

g) entitlements;

h) accompanying items (e.g. animals, bicycles);

i) related services (e.g. parking);

Product elements (with examples):

j) charging

    1) flat fare per ride;

    2) flat fare per time period;

    3) time based fare;

    4) distance based fare;

    5) time / distance based fare;

    6) means of transport discount / surcharge;

    7) time of day discounts / surcharges;

    8) transfer discounts ;

    9) frequency of use discounts;

    10) volume based discounts

        i) loyalty bonuses;

        ii) loyalty discounts;

        iii) surcharges;

        iv) penalties;

        v) related service charges;

k) geography

   1) graduated tickets;

   2) point-to-point tickets;

   3) zone based tickets;

   4) area based tickets;

l) time of day

   1) specific departure;

   2) period of the day;

   3) after or before a specific time;

   4) on a service arriving/departing after/before a specified time;

m) period

   1) expiry dates;

   2) date validity;

n) party size

   1) specified holder;

   2) anonymous individual;

   3) number of passengers by class;

o) concession status/entitlement;

p) others

   1) class of travel;

   2) reservations;

   3) additional features (parking and meals).

E.6 Tariff system independence

It is assumed within the standard that the tariff system is commercially neutral. Instead of mapping tariff systems, the basic elements contained in known tariff models and necessary for automated fare look-up/calculation should be available and applied individually within a specific transport product. Thus, the application will be able to make all necessary data available to the background system via the acceptance technology for the purposes of settlement and revenue apportionment.

E.7 Back office requirements

There needs to be a capability to extract data appropriate to the revenue-sharing and statistical requirements of product owners. The standards 24014 (IFM) addresses back office issues.

E.8 Performance requirements

In a high throughput environment such as transport, performance is a significant factor. Attention needs to be paid to the performance requirements of the scheme when implementing systems according to the standard.

Annex F
(informative)

Examples

F.1 General

In the examples of the three tables below, the following mandatory “administrative” information is appended to complete each product Data Group as described in 6.9. It is shown here once and not repeated in the examples.

Label

Table F.1 — Example of a label

APPLICATION DATA ELEMENT	EN 1545 DATA ELEMENT	DATA TYPE
Owner	CompanyId	ReferenceNumber
ExpiryDate	EndDate	DateCompact
ProductId	ProductId	ReferenceIdentifier

Instance identifier

Table F.2 — Example of an instance identifier

APPLICATION DATA ELEMENT	EN 1545 DATA ELEMENT	DATA TYPE
UnblockInstanceNumber	UnblockInstanceNumber	SequenceNumber
Status	ProductStatus	StatusCode
VersionNumber	VersionNumber	BIT STRING
DataGroupSequenceNumber	SequenceNumber	INTEGER
SecurityId	SecurityVersion	SEQUENCE
Priority	Priority	INT1

Seal

Table F.3 — Example of a seal

APPLICATION DATA ELEMENT	EN 1545 DATA ELEMENT	DATA TYPE
Seal	Authenticator	OCTET STRING

F.2 Concession

F.2.1 Creation of Holder Id and Basic Entitlement Definition

Holder Id with entitlement needs creation. This may then be a free concession or a flat, distance-related or zonal fare requiring separate payment by cash or STR.

Table F.4 — Concession; creation of holder ID and entitlement

HOLDER DATA GROUP - CONCESSION ID		
Data element	**Example**	**Datatype**
A-HOLDER PERSONAL DETAIL		
HolderSurname	Bloggs	Name
HolderForename	Josephine	Name
HolderBirthName	Biggles	Name
HolderBirthDate	21/04/33	Datef
HolderIdNumber	567890	ReferenceIdentifier
HolderCountryAlpha	UK	PrintabelString
HolderAddress	10 Acacia Avenue	Address
HolderZipcode	PR3 6JH	ReferenceIdentifier
HolderTelephone	01772 678954	NetworkAccess
HolderMobile	07945 678432	NetworkAccess
HolderLanguage	English	CHOICE
HolderIdDocumentNumber	NI: WZ 67 47 18A	ReferenceIdentifier
A-CUSTOMER ENTITLEMENT DETAIL		
EntitlementType	Free	ENUMERATED
EntitlementPassengerTotal	1 (or could be 2 if valid with Companion with extra A-Customer Entitlement Detail structure defined)	Quantity
EntitlementSerialNumber	987654	ReferenceNumber
EntitlementPrice	GBP 0	SEQUENCE
EntitlementReference	345678 (ie free in Preston, half-fare in Lancashire code – also gives reimbursement area of residence)	ReferenceIdentifier
EntitlementDiscount	DiscountType	INTM

Entitlement needs to define ‘free’ or ‘proportion of fare’ that the holder is entitled to (eg flat fare and amount, or proportion of fare eg 50 % and rounding rules eg up/down and whether to nearest 1p or to nearest 5p).

Entitlement Reference can define ‘area validity of the concession’ if specified at holder/entitlement level. (Alternatively it could be created at ‘Ticket’ level using A- Ticket Geography).

Under data protection regulations specific regard should be given to the accessibility of user details. In practice in the UK these details would be held in the back-office only and not on the card, but are shown in the example on the card for illustrative purposes.

F.2.2 Creation of concession period validity

The ticket needs to have further validity clarifications.

Table F.5 — Concession: creation of validity

TICKET PRODUCT DATA GROUP - TICKET VALIDITY		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	1234567	ReferenceNumber
TicketPrice	GBP 10	SEQUENCE
A TICKET VALIDITY INFORMATION		
ValidityRemoveDate	31/03/04	DateCompact
ValiditySaleDateTimeStamp	15/05/03, 10:37	INT3
ValidityNumberofAdults	1	Quantity
ValidityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING
ValiditySerialNumber	565656	ReferenceNumber
ValidityContractDependencyPointer	Pointer to Holder+Entitlement	InstancePointer
ValidityText	Free Concession	Databin

F.2.3 Use of concession

The concession entitlement may need to be logged on use.

Table F.6 — Concession: use of concession

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	1234567	ReferenceNumber
TicketPrice	GBP 10	SEQUENCE
A TICKET VALIDITY INFORMATION		
ValidityRemoveDate	31/03/04	DateCompact
ValiditySaleDateTimeStamp	15/05/03, 10:37	INT3
ValidityNumberofAdults	1	Quantity
ValidityNumberofChildren	0	Quantity

Table F.6 (*continued*)

TICKET PRODUCT DATA GROUP		
Data Element	**Example**	**Datatype**
ValidityPaymentMeans	Cash	BIT STRING
ValiditySerialNumber	565656	ReferenceNumber
ValidityContractDependencyPointer	SequenceNumber	InstancePointer
ValidityText	Free Concession	Databin
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	1234	ENUMERATED
SavedEventSequenceNumber	34	SequenceNumber
SavedEventDateTimeStamp	15/05/03, 14:00	INT3
SavedEventResultCode	Bus Ride	BIT STRING
SavedEventDeviceSerialNumber	Bus 2507 eg machine 456789	ReferenceNumber
SavedEventServiceOperator	Stagecoach	ReferenceNumber
SavedEventJourneyOrigin	Fare Stage 01 (Preston Bus Station)	SEQUENCE
SavedEventJourneyDestination	Fare Stage 4 (Deepdale Road)	SEQUENCE
L SECONDARY SAVED EVENT DATA (n)		
SavedEventStatusReceiptFlag	Yes	Flag
SavedEventJourneyRoute	35 (Servide)	ReferenceNumber
SavedEventJourneyRouteVariant	RouteVariant 1	ReferenceNumber
SavedEventJourneyRun	1400 (scheduled departure time or journey number)	ReferenceNumber
SavedEventDirection	Out	ENUMERATED
SavedEventPricePaid	GBP 0.00	SEQUENCE
SavedEventPaymentMeans	Pass	BIT STRING
SavedEventNonDiscountedPrice	GBP 00,80	SEQUENCE

The price paid and non-discounted price are both required to collect fare-foregone ie the amount due to the operator but not paid by the passenger. In this case the operator is still due the full fare of UK£00,80 because the passenger paid UK£0,00 (i.e. free). This may or may not be coded on the card (but would require a transaction record to be sent to the back office).

The fare foregone can be complex if rounding takes place, e.g.:

1) Fare 80p, half fare, round down to nearest 5p, pays 40p, operator due 40p;

2) Fare 85p, half fare, round down to nearest 5p, pays 40p, operator due 45p;

3) Fare 85p, half fare, round down to nearest 1p, pays 42p, operator due 43p.

NOTE The non-discounted fare paid may be subject to further reduction in the reimbursement calculation to allow for generated travel, but this is an IFM issue.

F.3 Carnet/Mehrfahrtenkarte (multi-ride, fixed price per ride)

F.3.1 The customer buys a "carnet" with cash

Table F.7 — Carnet: customer purchases the carnet

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	987654	ReferenceNumber
ticketPrice	£1,50	SEQUENCE
A TICKET VALIDITY INFORMATION		
validityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	14/05/03, 08:00	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
validityPaymentMeans	Cash	BIT STRING
validitySerialNumber	12345678	referencenumber
validityText	Smartlink Adult	Databin
L RIDE BASED TRAVEL LOAD		
RbtDateTimeStamp	14/05/03, 08:00	INT3
RbtSamSerialNumber	1234567890	referencenumber
RbtRetailerIdentifier	001 (Baileys Kiosk)	ReferenceNumber
RbtCouponsLoaded	10	Quantity
RbtLoadPrice	£8,50	SEQUENCE
RbtAfterLoadBalance	10	Signedamount

F.3.2 Boarding the vehicle the passenger validates one journey for their current ride

An L record is added for the first ride.

Table F.8 — Carnet: a journey is made

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	987654	ReferenceNumber
ticketPrice	£1,50	SEQUENCE
A TICKET VALIDITY INFORMATION		
validityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	14/05/03, 08:00	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
validityPaymentMeans	Cash	BIT STRING
validitySerialNumber	12345678	referencenumber
validityText	Smartlink Adult	Databin
L RIDE BASED TRAVEL LOAD		
RbtDateTimeStamp	14/05/03, 08:00	INT3
RbtSamSerialNumber	1234567890	referencenumber
RbtRetailerIdentifier	001 (Baileys Kiosk)	ReferenceNumber
RbtCouponsLoaded	10	Quantity
RbtLoadPrice	£8,50	SEQUENCE
RbtAfterLoadBalance	10	Signedamount
L RIDE BASED TICKET USAGE		
RbtCouponsDeducted	1	Quantity
RbtCountOfCoupons	9	Counter
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Usage	Enumerated
SavedEventSequenceNumber	35	SequenceNumber
SavedEventDateTimeStamp	14/05/03, 08:55	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	124935	ReferenceNumber
SavedEventServiceOperator	Citybus eg 0987654321	ReferenceNumber
SavedEventJourneyOrigin	Service 82 Stage 20	SEQUENCE
SavedEventJourneyDestination	Service 82 Stage 2	SEQUENCE

F.3.3 Further journey consumed

L records may be overwritten on each ride.

Table F.9 — Carnet: a further journey is made

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	987654	ReferenceNumber
ticketPrice	£1,50	SEQUENCE
A TICKET VALIDITY INFORMATION		
validityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	14/05/03, 08:00	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
validityPaymentMeans	Cash	BIT STRING
validitySerialNumber	12345678	Referencenumber
validityText	Smartlink Adult	Databin
L RIDE BASED TRAVEL LOAD		
RbtDateTimeStamp	14/05/03, 08:00	INT3
RbtSamSerialNumber	1234567890	Referencenumber
RbtRetailerIdentifier	001 (Baileys Kiosk)	ReferenceNumber
RbtCouponsLoaded	10	Quantity
RbtLoadPrice	£8,50	SEQUENCE
RbtAfterLoadBalance	10	Signedamount
L RIDE BASED TICKET USAGE		
RbtCouponsDeducted	1	Quantity
RbtCountOfCoupons	8	Counter
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Usage	Enumerated
SavedEventSequenceNumber	44	SequenceNumber
SavedEventDateTimeStamp	14/05/03, 17:45	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	99383	ReferenceNumber
SavedEventServiceOperator	Citybus eg 0987654321	ReferenceNumber

Table F.9 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
SavedEventJourneyOrigin	Service 82 Stage 2	SEQUENCE
SavedEventJourneyDestination	Service 82 Stage 20	SEQUENCE

In this case the L records overwrite the previous journey but a user could choose to keep all copies of L records or n copies by treating n L records in a cyclic manner.

F.3.4 Further top-up of rides

Part way through use of the 10 journeys (ie after 2 journeys), the carnet is topped up with 20 new rides, giving 18 left in total.

Table F.10 — Carnet: top-up of rides

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	987654	ReferenceNumber
TicketPrice	£1,50	SEQUENCE
A TICKET VALIDITY INFORMATION		
ValidityRemoveDate	0 (no expiry)	Datecompact
ValiditySaleDateTimeStamp	14/05/03, 08:00	INT3
ValidityPassengerClass	Adult	Enumerated
ValidityNumberofAdults	1	Quantity
ValidityNumberofChildren	0	Quantity
validityPaymentMeans	Cash	BIT STRING
validitySerialNumber	12345678	Referencenumber
validityText	Smartlink Adult	Databin
L RIDE BASED TRAVEL LOAD		
RbtDateTimeStamp	14/05/03, 08:00	INT3
RbtSamSerialNumber	1234567890	Referencenumber
RbtRetailerIdentifier	001 (Baileys Kiosk)	ReferenceNumber
RbtCouponsLoaded	10	Quantity
RbtLoadPrice	£8,50	SEQUENCE

Table F.10 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
RbtAfterLoadBalance	10	Signedamount
L RIDE BASED TRAVEL USAGE		
RbtCouponsDeducted	1	Quantity
RbtCountOfCoupons	7	Counter
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Usage	Enumerated
SavedEventSequenceNumber	56	SequenceNumber
SavedEventDateTimeStamp	14/05/03, 17:45	INT3
SavedEventResultCode	Ok	Bit String
SavedEventDevice	2215 (Bus No)	Deviceidentifier
SavedEventServiceOperator	Citybus eg 0987654321	ReferenceNumber
SavedEventJourneyOrigin	Service 82 Stage 2	SEQUENCE
SavedEventJourneyDestination	Service 82 Stage 20	SEQUENCE
L RIDE BASED TRAVEL LOAD		
RbtDateTimeStamp	14/05/03, 18:00	INT3
RbtSamSerialNumber	67676767	Referencenumber
RbtRetailerIdentifier	456789 (Citybus TVM)	ReferenceNumber
RbtCouponsLoaded	20	Quantity
RbtLoadPrice	£17,00	SEQUENCE
RbtAfterLoadBalance	27	Signedamount

F.4 Check-in/check-out using stored travel rights

F.4.1 Check-in

There are stored travel rights (transport tokens) previously loaded which can be used for the journey.

Table F.11 — Check in/ Check out: Stored Travel Rights availability

STORED TRAVEL RIGHTS DATA GROUP		
Data element	**Example**	**Datatype**
M STORED TRAVEL RIGHTS		

Table F.11 (*continued*)

STORED TRAVEL RIGHTS DATA GROUP		
Data element	**Example**	**Datatype**
strSamSerialNumber	8989771	ReferenceNumber
strUnit	Eurocents X 10	OCTET STRING
strMaxValue	100	Amount
L STORED TRAVEL RIGHTS BALANCE		
StrValue	100	Signedamount
L STORED TRAVEL RIGHTS LOAD		
StrDateTimeStamp	03/01/03, 14:30	INT3
StrSamSerialNumber	12345678	Referencenumber
StrRetailerId	Köln-Hauptbahnhof	ReferenceNumber
StrAmountLoaded	100	Amount
StrAfterLoadBalance	100	Signedamount
StrReceiptFlag	Yes	Flag

Table F.12 — Check in / Check out: Check In

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	8989771	ReferenceNumber
TicketPrice	-	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSequenceNumber	95	SequenceNumber
SavedEventdateTimeStamp	12/02/03, 14:56	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	565765676	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin	Köln-Hauptbahnhof	SEQUENCE
SavedEventJourneyDestination	Bonn	SEQUENCE

F.4.2 Check-out

Information about the check-out is stored by extending the ticket data group. The check-in and check-out information will be saved for a network specific period and might be used for calculating future ticket prices (e.g. reduction dependent on the number of journeys per day). From the saved event data a price for the ticket is calculated by the terminal and written to complete the ticket.

Table F.13 — Check in/ Check out: Check out

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	8989771	ReferenceNumber
TicketPrice	1,50 Euro	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSequenceNumber	95	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 14:56	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	565765676	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin	Köln-Hauptbahnhof	SEQUENCE
SavedEventJourneyDestination	Bonn	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-out	Enumerated
SavedEventSequenceNumber	102	SequenceNumber
SavedEventdateTimeStamp	12/02/03, 15:30	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	3242w43342	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin		SEQUENCE
SavedEventJourneyDestination	Köln-Sürth	SEQUENCE

The primary saved event structures may be limited in number and treated as a cyclic file once the limit has been reached, that is the oldest overwritten.

Stored travel rights which are to be used to pay for the ticket are updated.

Table F.14 — Check in/ Check out: Stored Travel Rights usage

STORED TRAVEL RIGHTS DATA GROUP		
Data element	**Example**	**Datatype**
M STORED TRAVEL RIGHTS		
strSamSerialNumber	8989771	ReferenceNumber
StrUnit	Eurocents X 10	OCTET STRING
StrMaxValue	100	Amount
L STORED TRAVEL RIGHTS BALANCE		
StrValue	85	Signedamount
L STORED TRAVEL RIGHTS LOAD		
StrDateTimeStamp	03/01/03, 14:30	INT3
StrSamSerialNumber	12345678	referencenumber
StrRetailerId	Köln-Hauptbahnhof	ReferenceNumber
StrAmountLoaded	100	Amount
StrAfterLoadBalance	100	Signedamount
StrReceiptFlag	Yes	Flag
L STR/CTA USAGE		
(str/cta)FareDeducted	15 STR = € 1,5	Amount

F.4.3 The journey continues

Information about further check-ins and check-outs are stored as the journey continues. The example assumes that all events are stored as new structures, that is, the L saved event structures are not treated cyclically. In this example, several check-in, check-out events are stored in one ticket data group. It would be equally possible to delete the first complete journey ticket and write a new one for later journeys, however this would not allow the possibility to generate discounts based upon the number of journeys in a day. Note that the ticket price is updated to show the price for the two journeys under this ticket.

Table F.15 — Check in/ Check out: the journey continues

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	8989771	ReferenceNumber
TicketPrice	2,50 Euro	SEQUENCE

Table F.15 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSequenceNumber	95	SequenceNumber
SavedEventTime	12/02/03, 14:56	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	565765676	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin	Köln-Hauptbahnhof	SEQUENCE
SavedEventJourneyDestination	Bonn	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-out	Enumerated
SavedEventSequenceNumber	102	I SequenceNumber
SavedEventDateTimeStamp	12/02/03, 15:30	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	3242w43342	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin		SEQUENCE
SavedEventJourneyDestination	Köln-Sürth	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSequenceNumber	110	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 17:30	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	3298w43342	ReferenceNumber
SavedEventServiceOperator	SWB	ReferenceNumber
SavedEventJourneyOrigin	Köln-Sürth	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-out	Enumerated
SavedEventSequenceNumber	113	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 18:00	INT3

Table F.15 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	1142w43342	ReferenceNumber
SavedEventServiceOperator	SVB	ReferenceNumber
SavedEventJourneyDestination	Bonn HBF	SEQUENCE

The stored travel rights balance will be updated accordingly.

Table F.16 — Check in/ Check out: further Stored Travel Rights usage

STORED TRAVEL RIGHTS DATA GROUP		
Data Element	**Example**	**Datatype**
M STORED TRAVEL RIGHTS		
StrSamSerialNumber	8989771	ReferenceNumber
StrUnit	Eurocents X 10	OCTET STRING
StrMaxValue	100	Amount
L STORED TRAVEL RIGHTS BALANCE		
StrValue	75	Signedamount
L STORED TRAVEL RIGHTS LOAD		
StrDateTimeStamp	03/01/03, 14:30	INT3
StrSamSerialNumber	12345678	referencenumber
StrRetailerId	Köln-Hauptbahnhof	ReferenceNumber
StrAmountLoaded	100	Amount
StrAfterLoadBalance	100	Signedamount
StrReceiptFlag	Yes	Flag
L STR/CTA USAGE		
(str/cta)FareDeducted	15 STR = € 1,5	Amount
L STR/CTA USAGE		
(str/cta)FareDeducted	10 STR = € 1,0	Amount

F.5 Be-in/Be-out

F.5.1 After Boarding

After boarding the vehicle and leaving the stop the presence of the media in the vehicle is detected at which time it needs to be determined if post payment (charge to account) is applicable.

Table F.17 — Be in/ be out: entitlement to ride

CUSTOMER ENTITLEMENT DATA GROUP		
Data element	**Example**	**Datatype**
M-CUSTOMER ENTITLEMENT		
entitlementSamSerialNumber	111222	ReferenceNumber
EntitlementType	Post Paid Entitlement	Enumerated
entitlementDossierReference	12345	Referenceidentifier
A-CTA ADDITIONAL INFO		
CtaTransactionLimit	20 Euro	SEQUENCE
CtaAccountReference	99876	Referenceidentifier

If permitted, a ticket is written.

Table F.18 — Be in / be out: after boarding

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	9988877	ReferenceNumber
TicketPrice	-	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Be-in	Enumerated
SavedEventSequenceNumber	27	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 14:53	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	565765676	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin	Köln-Hauptbahnhof	SEQUENCE

F.5.2 The journey continues

As the journey continues after each stop the presence of the media in the vehicle is detected and further data structures may be written. Calculating and charging the fare in the background system is possible without writing this information to the media.

The reason for writing the be In records to the card when the actual price calculation is carried out after the event in the back office is twofold:

— inspection in the vehicle;

— fare calculation in subsequent journeys.

Table F.19 — Be in / be out: the journey continues

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	9988877	ReferenceNumber
TicketPrice	-	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Be-in	Enumerated
SavedEventSequenceNumber	27	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 14:53	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	565765676	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin	Köln-Hauptbahnhof	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Be-in	Enumerated
SavedEventSequenceNumber	27	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 15:30	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	3242w43342	ReferenceNumber
SavedEventServiceOperator	KVB	ReferenceNumber
SavedEventJourneyOrigin		SEQUENCE
SavedEventJourneyDestination		Enumerated

Table F.19 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Be-in	Enumerated
SavedEventSequenceNumber	30	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 17:30	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	3298w43342	ReferenceNumber
SavedEventServiceOperator	SWB	ReferenceNumber
SavedEventJourneyOrigin	Köln-Sürth	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Be-in	Enumerated
SavedEventSequenceNumber	31	SequenceNumber
SavedEventDateTimeStamp	12/02/03, 18:53	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	1142w43342	ReferenceNumber
SavedEventServiceOperator	SWB	ReferenceNumber

F.6 Streifenkarte

F.6.1 Buying the Streifenkarte for cash

Table F.20 — Streifenkarte: purchasing for cash

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	767676	ReferenceNumber
ticketPrice	10 Euro	SEQUENCE
A TICKET VALIDITY INFORMATION		
ValiditySaleDateTimeStamp	12/04/03, 17:49	INT3
ValidityPassengerClass	Adult	Enumerated

Table F.20 (*continued*)

TICKET PRODUCT DATA GROUP		
Data Element	**Example**	**Datatype**
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
validityPaymentMeans	Cash	BIT STRING
validitySerialNumber	73487589342759	Referencenumber
validityText	„Mehrfahrtenkarte“	Databin
L RIDE BASED TRAVEL LOAD		
RbtDateTimeStamp	12/04/03, 17:49	INT3
RbtSamSerialNumber	123456789	Referencenumber
RbtRetailerIdentifier	45675	ReferenceNumber
RbtCouponsLoaded	20	Quantity
RbtLoadPrice	10 Euro	SEQUENCE
RbtAfterLoadBalance	120	Signedamount

F.6.2 Boarding the vehicle

Two coupons are validated/used.

Table F.21 — Streifenkarte: boarding the vehicle

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	767676	ReferenceNumber
ticketPrice	10 Euro	SEQUENCE
A TICKET VALIDITY INFORMATION		
validitySaleDateTimeStamp	12/04/03, 17:49	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
validityPayMethod	Cash	BIT STRING
validitySerialNumber	73487589342759	referencenumber
validityText	„Mehrfahrtenkarte“	Databin

Table F.21 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
L RIDE BASED TICKET LOAD		
RbtDateTimeStamp	12/04/03, 17:49	INT3
RbtTime	17:49	Timecompact
RbtSamSerialNumber	123456789	referencenumber
RbtRetailerIdentifier	45675	ReferenceNumber
RbtCouponsLoaded	20	Quantity
RbtLoadPrice	10 Euro	SEQUENCE
RbtAfterLoadBalance	120	Signedamount
L RIDE BASED TICKET USAGE		
RbtCouponsDeducted	2	Quantity
RbtCountOfCoupons	18	Counter
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Validation	Enumerated
SavedEventSequenceNumber	15	SequenceNumber
SavedEventDateTimeStamp	14/05/03, 18:10	INT3
SavedEventResultCode	Validated	Bit String
SavedEventDeviceSerialNumber	1232354	ReferenceNumber
SavedEventServiceOperator	2374823789	ReferenceNumber
SavedEventJourneyOrigin	Köln-Friesneplatz	SEQUENCE
SavedEventJourneyDestination	Köln-Hauptbahnhof	SEQUENCE

F.6.3 Further journeys

Further structures are appended to the ticket on each ride. The ticket may be deleted when the count of coupons becomes 0 subject to the validity information on the ticket and business rules prevailing.

Table F.22 — Streifenkarte: further journeys

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	767676	ReferenceNumber

Table F.22 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
ticketPrice	10 Euro	SEQUENCE
A TICKET VALIDITY INFORMATION		
validitySaleDateTimeStamp	12/04/03, 17:49	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
validityPayMethod	Cash	BIT STRING
validitySerialNumber	73487589342759	Referencenumber
validityText	„Mehrfahrtenkarte“	Databin
L RIDE BASED TICKET LOAD		
RbtDateTimeStamp	12/04/03, 17:49	INT3
RbtTime	17:49	Timecompact
RbtSamSerialNumber	123456789	Referencenumber
RbtRetailerIdentifier	45675	ReferenceNumber
RbtCouponsLoaded	20	Quantity
RbtLoadPrice	10 Euro	SEQUENCE
RbtAfterLoadBalance	120	Signedamount
L RIDE BASED TICKET USAGE		
RbtCouponsDeducted	2	Quantity
RbtCountOfCoupons	18	Counter
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Validation	Enumerated
SavedEventSequenceNumber	15	SequenceNumber
SavedEventDateTimeStamp	14/05/03, 18:10	INT3
SavedEventResultCode	Validated	Bit String
SavedEventDeviceSerialNumber	1232354	ReferenceNumber
SavedEventServiceOperator	2374823789	ReferenceNumber
SavedEventJourneyOrigin	Köln-Friesneplatz	SEQUENCE

Table F.22 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
SavedEventJourneyDestination	Köln-Hauptbahnhof	SEQUENCE
L RIDE BASED TICKET USAGE		
RbtCouponsDeducted	4	Quantity
RbtCountOfCoupons	16	Counter
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Validation	Enumerated
SavedEventSequenceNumber	18	SequenceNumber
SavedEventDateTimeStamp	14/04/03, 14:10	INT3
SavedEventResultCode	Validated	Bit String
SavedEventDeviceSerialNumber	9232354	ReferenceNumber
SavedEventServiceOperator	9374823789	ReferenceNumber
SavedEventJourneyOrigin	Bonn-Nord	SEQUENCE
SavedEventJourneyDestination	Bonn-Bad Godesberg	SEQUENCE

F.7 Rail travel with reservation

F.7.1 Reservation

A ticket is created.

Table F.23 — Rail travel: reservation

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	886644	ReferenceNumber
TicketPrice	100 Euro	SEQUENCE
A TICKET VALIDITY INFORMATION		
validitySaleDateTimeStamp	12/04/03, 17:49	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity

Table F.23 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
validityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING
validitySerialNumber	73487589300003	Referencenumber
validityText	Flexible one way ticket	Databin
A RESERVATION BASIC		
reservationServiceId	12345	referenceidentifier
ReservationDepartureDateTimeStamp	17/04/03, 09:15	INT3
ReservationCoachId	C	referenceNumber
ReservationSeatNumberFrom	27	ReferenceNumber
ReservationPlaceType	UIC 920-16 code	Enumerated
A-TICKET GEOGRAPHY		
ticketJourneyOrigin	Paris	LocationIdentifier
ticketJourneyDestination	Rome	LocationIdentifier

F.7.2 The journey is made

Table F.24 — Rail travel: a journey is made

TICKET PRODUCT DATA GROUP		
Data Element	**Example**	**Datatype**
M TICKET		
ticketSamSerialNumber	886644	ReferenceNumber
TicketPrice	100 Euro	SEQUENCE
A TICKET VALIDITY INFORMATION		
validitySaleDateTimeStamp	12/04/03, 17:49	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING

Table F.24 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
validitySerialNumber	73487589300003	Referencenumber
validityText	Flexible one way ticket	Databin
A RESERVATION BASIC		
reservationServiceId	12345	referenceidentifier
ReservationDepartureDateTimeStamp	17/04/03, 09:15	INT3
ReservationCoachId	C	referenceNumber
ReservationSeatNumberFrom	27	ReferenceNumber
ReservationPlaceType	UIC 920-16 code	Enumerated
A-TICKET GEOGRAPHY		
ticketJourneyOrigin	Paris	LocationIdentifier
ticketJourneyDestination	Rome	LocationIdentifier
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Gate Entry	Enumerated
SavedEventSequenceNumber	11	SequenceNumber
SavedEventDateTimeStamp	17/04/03, 09:00	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	98769	ReferenceNumber
SavedEventServiceOperator	9890803883983	ReferenceNumber
SavedEventJourneyOrigin	Paris GdL	SEQUENCE
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Gate Exit	Enumerated
SavedEventSequenceNumber	12	SequenceNumber
SavedEventDateTimeStamp	17/04/03, 19:00	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	78654	ReferenceNumber
SavedEventServiceOperator	125678544355	ReferenceNumber
SavedEventJourneyDestination	Rome	SEQUENCE

F.8 RET[x)] Multi-leg return journey

F.8.1 General

Some important usage rules of the RET return ticket in the public transport system:

— usage of the ticket is restricted to transportation with RET;

— ticket needs to be validated at check-in;

— after checking in, a single journey can be made, which can consist of several journey legs. The maximum travelling time for a single journey is 1,25 h;

— transfer rule 1: all consecutive journey legs will be considered to be part of a single journey, on condition that the maximum transfer time between the journey legs will be 35 min;

— transfer rule 2: at every transfer, check-out and check-in are obligatory;

— transfer rule 3: when the time period between two journey legs of the journey out is more than 35 min, the system considers the respective ride as the (first journey leg of the) journey back.

The example below does not show all the complexity of the usage rules (e.g. restrictions on time). In any event, it is likely that the complex calculations with regard to the ticket will be done by the terminal, not the card.

F.8.2 The customer buys his return ticket with cash

Table F.25 — RET: a ticket is purchased

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	111111	ReferenceNumber
ticketPrice	€ 1,10	SEQUENCE
A RETAILER		
RetailerIdentifier	123 (RET, CS)	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	ReferenceIdentifier
A TICKET VALIDITY INFORMATION		
validityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	01/09/03, 09:20	INT3
ValidityPassengerClass	Adult	Enumerated

x) RET – Rotterdamse Electrische Tramweg (Netherlands).

validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity

Table F.25 (*continued*)

TICKET PRODUCT DATA GROUP		
Data Element	**Example**	**Datatype**
validityPaymentMeans	Cash	BIT STRING
ValidityTariffId	27	ReferenceNumber
ValiditySerialNumber	12345678	Referencenumber
ValidityText	RET retour volw.	Databin

NOTE Payment can also be made with a credit/debit card or an e-purse.

F.8.3 The customer checks in

Table F.26 — RET: Check in

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	111111	ReferenceNumber
ticketPrice	€ 1,10	SEQUENCE
A RETAILER		
RetailerIdentifier	123 (RET, CS)	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	ReferenceIdentifier
A TICKET VALIDITY INFORMATION		
validityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	01/09/03, 09:20	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING
ValidityTariffId	27	ReferenceNumber
ValiditySerialNumber	12345678	Referencenumber
ValidityText	RET retour volw.	Databin

Table F.26 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
L PRIMARY SAVED EVENT DATA (1)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	31	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 09:36	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	12345	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	123 (RET metro CS)	Sequence
SavedEventJourneyDestination	-	Sequence

Note that there is no destination information yet, because this information will be generated during check-out.

F.8.4 The customer checks out

Table F.27 — RET: Check out

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	111111	ReferenceNumber
TicketPrice	€ 1,10	SEQUENCE
A RETAILER		
RetailerIdentifier	123 (RET, CS)	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	ReferenceIdentifier
A TICKET VALIDITY INFORMATION		
validityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	01/09/03, 09:20	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity

Table F.27 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
ValidityPaymentMeans	Cash	BIT STRING
ValidityTariffId	27	ReferenceNumber
ValiditySerialNumber	12345678	Referencenumber
ValidityText	RET retour volw.	Databin
L PRIMARY SAVED EVENT DATA (1)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	31	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 09:36	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	12345	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	123 (RET metro CS)	Sequence
SavedEventJourneyDestination	-	Sequence
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-out	Enumerated
SavedEventSeqenceNumber	32	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 09:54	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	23456	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	123 (RET metro CS)	SEQUENCE
SavedEventJourneyDestination	234 (RET metro Coolhaven)	Sequence

NOTE In this example, the logged “Primary Saved Event Data” of the check-in are not overwritten by the same data with regard to the check-out since the first check-in time is essential for the ticket validity (1,25 h valid from first check-in time on) and should be known by the system.

Subsequent check-in and check-out event will overwrite one another to minimise space utilisation, although if space is not a problem, all records may be kept. Alternatively, a cyclic set of “n” saved event records may be kept such that the resulting ticket shows the first check-in and the most recent “n” check-in and check-out events.

F.8.5 The journey continues: Check-in of next journey leg

Table F.28 — RET: Check in next leg

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	111111	ReferenceNumber
TicketPrice	€ 1,10	SEQUENCE
A RETAILER		
RetailerIdentifier	123 (RET, CS)	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	ReferenceIdentifier
A TICKET VALIDITY INFORMATION		
ValidityRemoveDate	0 (no expiry)	Datecompact
ValiditySaleDateTimeStamp	01/09/03, 09:20	INT3
ValidityPassengerClass	Adult	Enumerated
ValidityNumberofAdults	1	Quantity
ValidityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING
ValidityTariffId	27	ReferenceNumber
ValiditySerialNumber	12345678	Referencenumber
ValidityText	RET retour volw.	Databin
L PRIMARY SAVED EVENT DATA (1)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	31	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 09:36	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	12345	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	123 (RET metro CS)	Sequence
SavedEventJourneyDestination	-	Sequence

Table F.28 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	41	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 10:18	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	34567	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	234 (RET metro Coolhaven)	SEQUENCE
SavedEventJourneyDestination	-	Sequence

F.8.6 The journey continues: Check-out of next journey leg

Table F.29 — RET: Check out next leg

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	111111	ReferenceNumber
TicketPrice	€ 1,10	SEQUENCE
A RETAILER		
RetailerIdentifier	123 (RET, CS)	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	ReferenceIdentifier
A TICKET VALIDITY INFORMATION		
ValidityRemoveDate	0 (no expiry)	Datecompact
ValiditySaleDateTimeStamp	01/09/03, 09:20	INT3
ValidityPassengerClass	Adult	Enumerated
ValidityNumberofAdults	1	Quantity
ValidityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING
ValidityTariffId	27	ReferenceNumber

Table F.29 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
ValiditySerialNumber	12345678	Referencenumber
ValidityText	RET retour volw.	Databin
L PRIMARY SAVED EVENT DATA (1)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	31	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 09:36	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	12345	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	123 (RET metro CS)	Sequence
SavedEventJourneyDestination	-	Sequence
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-out	Enumerated
SavedEventSeqenceNumber	42	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 10:44	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	45678	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	234 (RET metro Coolhaven)	SEQUENCE
SavedEventJourneyDestination	345 (RET metro Marconiplein)	Sequence

F.8.7 Return journey: Check-in

The system recognises that this is the return journey based upon the journey time limit of one and a quarter hours.

Table F.30 — RET: Check in return journey

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	111111	ReferenceNumber
ticketPrice	€ 1,10	SEQUENCE

Table F.30 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
A RETAILER		
RetailerIdentifier	123 (RET, CS)	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	ReferenceIdentifier
A TICKET VALIDITY INFORMATION		
validityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	01/09/03, 09:20	INT3
ValidityPassengerClass	Adult	Enumerated
validityNumberofAdults	1	Quantity
validityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING
ValidityTariffId	27	ReferenceNumber
ValiditySerialNumber	12345678	Referencenumber
ValidityText	RET retour volw.	Databin
L PRIMARY SAVED EVENT DATA (1)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	31	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 09:36	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	12345	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	123 (RET metro CS)	Sequence
SavedEventJourneyDestination	-	Sequence
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	55	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 15:48	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	56789	ReferenceNumber

Table F.30 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	345 (RET metro Marconiplein)	SEQUENCE
SavedEventJourneyDestination	-	Sequence

NOTE The new first check-in for the return journey is also kept in a similar manner to the first check-in for the outward leg. This enables the system to check the time limit rules for the return journey. The next saved event data will be stored as a third saved event record which will be overwritten by subsequent check-in and check-out events until the journey is completed, unless all records are being kept or a cyclic record keeping system is employed as previously described.

The journey continues as before.

F.8.8 Return journey: Check-out

Table F.31 — RET: Check out return journey

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketSamSerialNumber	111111	ReferenceNumber
TicketPrice	€ 1,10	SEQUENCE
A RETAILER		
RetailerIdentifier	123 (RET, CS)	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	ReferenceIdentifier
A TICKET VALIDITY INFORMATION		
ValidityRemoveDate	0 (no expiry)	Datecompact
validitySaleDateTimeStamp	01/09/03, 09:20	INT3
ValidityPassengerClass	Adult	Enumerated
ValidityNumberofAdults	1	Quantity
ValidityNumberofChildren	0	Quantity
ValidityPaymentMeans	Cash	BIT STRING
ValidityTariffId	27	ReferenceNumber

Table F.31 (*continued*)

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
ValiditySerialNumber	12345678	Referencenumber
ValidityText	RET retour volw.	Databin
L PRIMARY SAVED EVENT DATA (1)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	31	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 09:36	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	12345	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	123 (RET metro CS)	Sequence
SavedEventJourneyDestination	-	Sequence
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-in	Enumerated
SavedEventSeqenceNumber	55	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 15:48	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	56789	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	345 (RET metro Marconiplein)	SEQUENCE
SavedEventJourneyDestination	-	Sequence
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Check-out	Enumerated
SavedEventSeqenceNumber	56	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 16:03	INT3
SavedEventResultCode	OK	Bit String
SavedEventDeviceSerialNumber	56789	ReferenceNumber
SavedEventServiceOperator	RET	ReferenceNumber
SavedEventJourneyOrigin	345 (RET metro Marconiplein)	SEQUENCE
SavedEventJourneyDestination	123 (RET metro CS)	SEQUENCE

The journey continues until completion.

F.9 Imagin'R annual ticket in zonal fare scheme

F.9.1 Ticket sale

The ticket is issued. A primary saved event data structure is written to record the sale date and sale device.

Table F.32 — Zonal fare scheme: a ticket is purchased

TICKET PRODUCT DATA GROUP		
Data element	**Example**	**Datatype**
M TICKET		
TicketVersionSamSerialNumber	1	ReferenceNumber
TicketPrice	Price	SEQUENCE
A-VALIDITY PERIOD		
ValidityStartDate	01/10/03	Datecompact
ValidityEndDate	30/09/04	Datecompact
A RETAILER		
RetailerIdentifier	Retailer	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	Referenceidentifier
A-Ticket Geography		
TicketZoneMap	3456	INT3
L PRIMARY SAVED EVENT DATA		
SavedEventTypeCode	Sale	Enumerated
SavedEventSequenceNumber	23	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 15:48	INT3
SavedEventResultCode	OK	Bit String

F.9.2 The ticket is used

Table F.33 — Zonal fare scheme: the ticket is used

Ticket Product Data Group		
Data Element	**Example**	**Datatype**
M TICKET		
TicketVersionSamSerialNumber	1	ReferenceNumber
TicketPrice	Price	SEQUENCE
A-VALIDITY PERIOD		
ValidityStartDate	01/10/03	Datecompact
ValidityEndDate	30/09/04	Datecompact
A RETAILER		
RetailerIdentifier	Retailer	ReferenceNumber
RetailerSaleAgent	-	ReferenceNumber
RetailerSaleDevice	12345	Referenceidentifier
A-TICKET GEOGRAPHY		
TicketZoneMap	3456	INT3
L PRIMARY SAVED EVENT DATA		
SavedEventTypeCode	Sale	Enumerated
SavedEventSequenceNumber	23	SequenceNumber
SavedEventDateTimeStamp	01/09/03, 15:48	INT3
SavedEventResultCode	OK	Bit String
L PRIMARY SAVED EVENT DATA (n)		
SavedEventTypeCode	Validation	Enumerated
SavedEventSequenceNumber	30	SequenceNumber
SavedEventDateTimeStamp	10/10/03, 09:00	INT3
SavedEventResultCode	OK	Bit String

Annex G
(informative)

Accessing the Interoperable Public Transport Application

G.1 General

G.1.1 Introduction

This annex is provided to indicate how an instance of an application may be found on a variety of Machine Readable Cards presented to the terminal.

This task is made more complex because some of the most common card platforms now in widespread use do not comply with all parts of ISO/IEC 14443.

This section describes the details of activation of the Machine Readable Card and accessing the application.

G.1.2 Card and application activation

The current existing card types in different schemes are based on simple memory cards, secured memory cards, simple processor cards and highly sophisticated processor cards.

Starting with the identification of the card platform using a generic card driver for the ISO/IEC 14443 interface, the terminal loads successive platform specific drivers which translates the logical card profile into a command set and the data group profile into security related commands respectively.

The sequence of security and administrative commands depends on the application / business rules and is illustrated in Figure G.1.

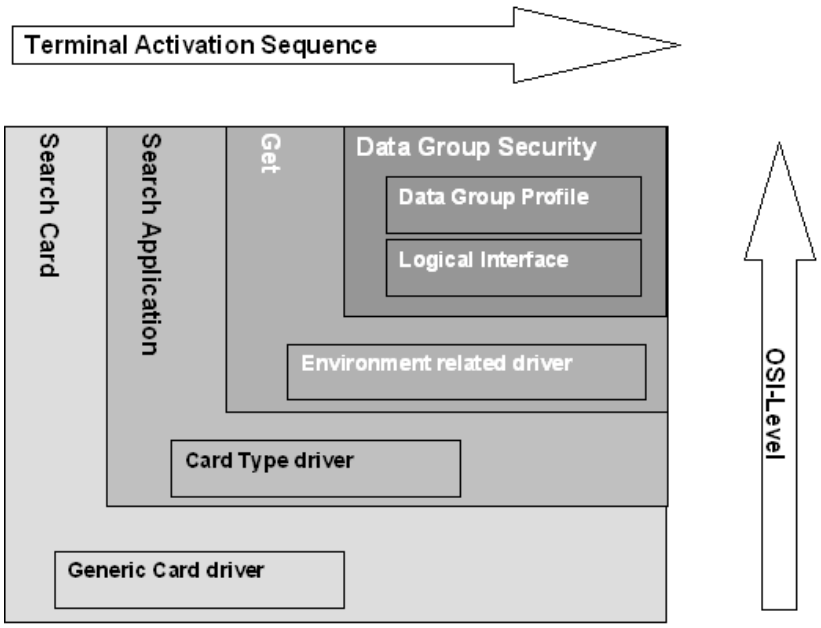

Figure G.1 — Card and application activation

G.2 Card type identification

G.2.1 General

A terminal shall be able to interpret the ATQ for all types of cards acceptable in the implementation of this European Standard.

This subclause covers the known popular card types and their responses according the different parts of ISO/IEC14443, as illustrated in Table G.1.

Table G.1 — Responses of known cards types

Card platforms				
ISO/IEC	Protocol Type			
14443-2	A		B	
14443-3	REQ A		REQ B	
	ATQA		ATQB	
	Proprietary / SAK	SAK	App / Protocol Info	Proprietary
14443-4	Proprietary	ATS	Answer to ATTRIB	Proprietary

G.2.2 Single or multi application processor cards, ISO/IEC 14443-4

Cards which are able to communicate according ISO/IEC 14443-4 should indicate the existance of an instance of the application in the ATS historical bytes (Type A) or in the specific application data (Type B). in order to make the activation sequence more efficient.

G.2.3 Small and secured memory cards

Offer proprietary information about the platform in the ATQA and SAK.

G.3 Identification of the application

G.3.1 General

Different card platforms available to an application usually need different logical access modes. After successful recognition of a specific card platform the terminal loads a particular card platform driver that provides the correct commands to access card data. With this approach it is then possible to use any card technology.

G.3.2 Single or multi-application processor cards, ISO/IEC 7816-4

Multi-application processor cards according ISO/IEC 14443-4 and ISO/IEC 7816-4 comprise standardised methods of identifying applications in the card. The historical bytes of the ATS (type A) may show the existence of an EF-DIR which contains information about the stored applications.

Another way of identifying the existence of an instance of the application is the standardised selection of an application with the select AID command according ISO/IEC 7816-4. This method may also be used if there is only one application (the Interoperable Public Transport Application) on the card or an EF-DIR is not supported.

G.3.3 Processor cards, not supporting ISO/IEC 7816-4

Processor cards which do not support ISO/IEC 7816-4 might comprise a proprietary method of identifying an instance of the application in the card. The terminal should identify the application after recognising the special card type using the offered selection method.

G.4 Identification of the profile

After identification of the card platform and the existence of the application the terminal has to identify the registered profile ID. The terminal stores an appropriate profile for each specific profile ID used in an instance of the application. The profile translates the abstract logical interface to the application and card specific commands.

G.5 Access to the product

After identification and selection of the correct profile, all the information necessary to access to the products is then available.

The last step necessary to gain access to all permitted data elements is to load the product/contract specific data group profile, which contains all security related information. It might also contain the specific structure of the product.

Annex H
(normative)

Relationship between legacy systems and the Interoperable Public Transport Application

H.1 General

A compliant application is one which meets the following criteria:

a) fully conforms to this European Standard. In the case of legacy systems, this may be achieved through mapping the legacy system into the application using the profile system of the application as described in Clause 9.

Or as a migration path

b) does not fully conform to this standard but is able to meet specific requirements

   1) is registered with the Interoperable Public Transport Application registration authority and

   2) has the ability to recognise an instance of the application and

   3) has the ability to recognise, handle and create a specific application product known as the application international product and

   4) has the ability to write and handle its own (local) products in an application existing as an interoperability extension to a foreign (non local) legacy application on a foreign (non local) card and

   5) has created an interoperability extension to its own local legacy application to allow an application international product and foreign (non local) legacy products to be written to the card carrying the local legacy application

where

c) an application international product is defined as meaning

   1) a single through ticket for international travel,

   2) a CTA product data group where the account link is to an international finance sector account (e.g. credit card account; or European bank account) and

   3) an STR product data group where an international agreement is in operation;

d) the local products of the legacy system

   1) are a full set or subset of the products available as selected by the local operator;

   2) are wrapped in an application wrapper data group as shown in Figure H.1.

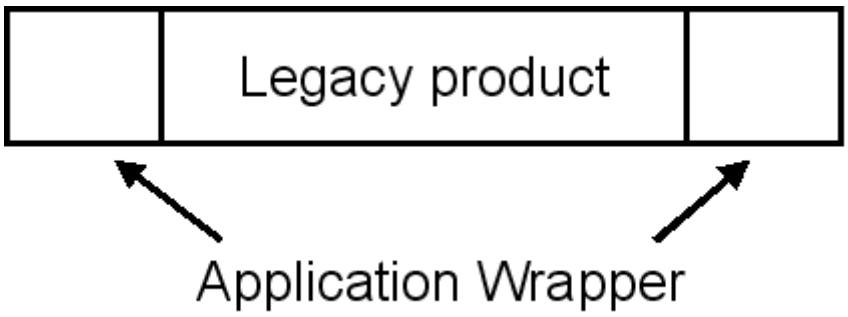

Figure H.1 — Application wrapper

H.2 Accessing compliant applications

A transport application on a card may be accessed in two ways:

— selection of the legacy application according to its normal procedures;

— selection of the application complaint part of the application, containing application compliant products (as defined above) by selecting the application according to Annex G.

The implementation of the methodology to support the selection of the compliant application is up to the local scheme to choose but will take one of the forms shown in Figure H.2, Figure H.3 and Figure H.4.

Figure H.2 shows the case of a ticket written with application (A) being used in the environment of application (B). Application (B) cannot access legacy application (A) directly but using an application compliant terminal, it can access the application international product and is able to write its products into the application environment using the wrapper data group.

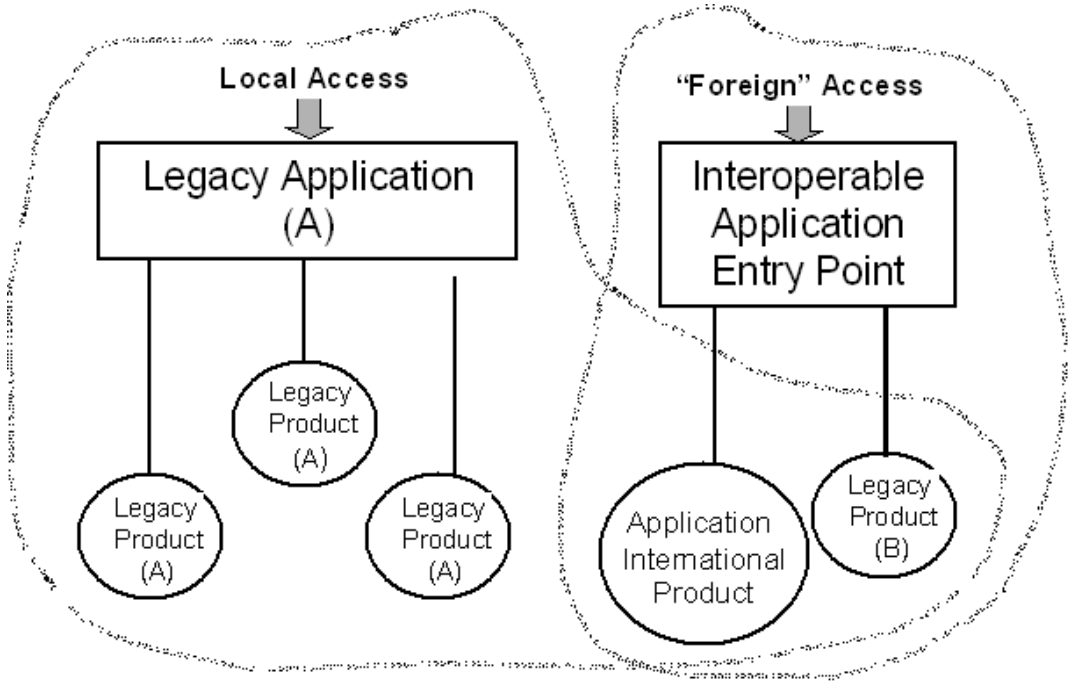

Figure H.2 — Inter-environment operation

The application compliant entry point may be created and accessed by means of adding a “stub” to the legacy application to allow the compliant entry point to be facilitated. The method of achieving this is implementer dependent.

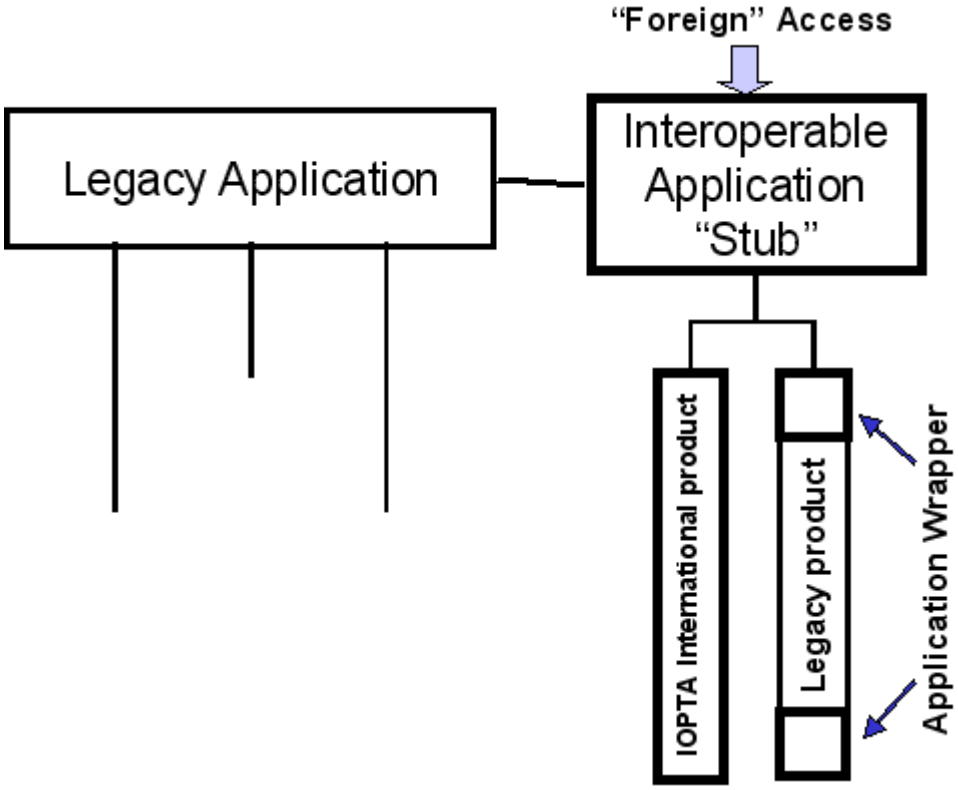

Figure H.3 — Interoperable Public Transport Application stub

The hierarchy of access to these applications is not relevant to the required access as shown in Figure H.4.

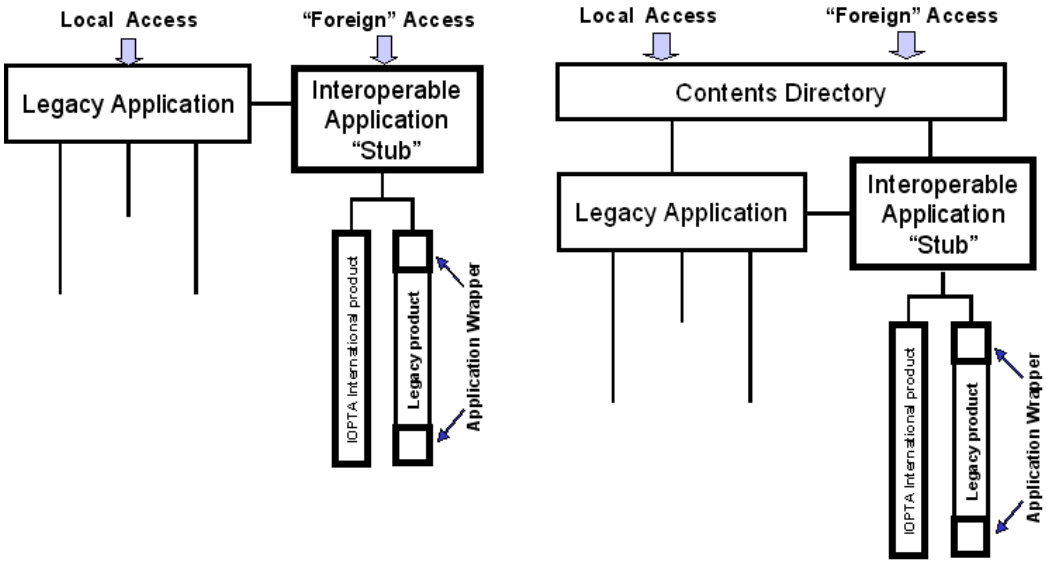

Figure H.4 — Hierarchy of access

Annex I
(informative)

Supporting Legacy Systems

I.0 Introduction

While the Interoperable Public Transport Application will suit those developing new systems or entirely replacing existing systems, it is recognised that for a considerable period of time legacy systems will remain current. In addition, where conversion to the application is a desirable goal, it will have to be achieved through a logical migration path rather than by overnight replacement, not least because many customers will hold tickets, passes and entitlements on the legacy system products. In this European Standard, these concepts are embodied in normative Annex H.

This informative annex discusses alternative approaches towards achieving a practical migration path.

I.1 The Requirement For Interoperability

The migration policy described here recognises that there are various stages of interoperability and that in a phased convergence policy it is important to tackle first only those that will have a major impact on the passenger or cardholder. These are seen as:

— for a traveller holding a card containing a transport application issued in one country, region or network to be able to travel to another country, region or Network and be able to use their card to have local tickets written to it such that he or she can travel without having to acquire a new, local card;for a traveller to have an international through ticket written to their card containing a transport application issued in one country, region or network, and be able to use that ticket without problem or other special action in all the countries, regions or Networks for which the ticket is valid.

I.2 Alternative approaches

The above requirements are embodied in Annex H and may be achieved in a number of ways, some of which are detailed below:

a) add an Interoperable Public Transport Application to cards containing a local legacy card and write the required tickets to it. The implication of this is that there is sufficient card memory space available and that point of service terminals (terminal) in all locations where the card is to be used are fully able to handle the application as well as their local application. During any transition period, this is unlikely to be the case;

b) map all participating legacy systems to one another such that each application is able to handle tickets written to foreign legacy transport applications. Work done on this approach shows it is possible but requires complex processing in terminals;

c) fully map legacy application to the application using the profile system as set down in Clause 9 such that they appear at the logical interface as full applications;

d) nominate and register legacy transport applications as “Compliant” applications, which means:

e) they are known to all other compliant applications;

f) they can handle a limited set (perhaps just one) of application ticket products;

g) they are able to read and write their own legacy products into an application structured environment.

I.3 Possible implementation methodologies

I.3.1 General

The above approaches may be implemented in concept as shown below:

I.3.2 The full Interoperable Public Transport Application implementation

This implementation assumes the card has sufficient space to hold both a legacy application and the application. It also assumes the terminal has sufficient capability to hold both the legacy and application software necessary to handle the respective card applications.

In this scenario, the legacy application and the Interoperable Public Transport Application are independent of one another and the application is fully compliant. This is clearly the most desirable scenario from a migration point of view, but it is also the most unlikely to be implemented due to card and terminal restrictions.

I.3.3 Legacy system mapping

This implementation implies the full mapping of one legacy application to another. If the application is used as the common interface format, then this may be classed as a stepping-stone to migration to it.

I.3.4 Mapping to the application using application profiles

In this case the legacy application can be fully mapped to the application using the application profiling system which means that the application on the card may be migrated without significant change to the terminal software. What is mainly required is support for the application profiling system.

I.3.5 The compliant application

This case assumes none of the above cases will apply and seeks to find a way to begin migration to Interoperable Public Transsport Application by meeting the mandatory requirements for compliance as set down in Annex G. To achieve this, the following is required:

a) legacy applications seeking to be recognised as compliant will register with an application registration authority responsible to CEN but not necessarily operated by CEN. This authority is to be defined and appointed in order to hold information containing unique scheme implementations and numbering schemes;

b) legacy applications need to enhance their systems to enable a second access point into the transport application on the card, which is the application compliant access point;

c) The application compliant access point needs to deliver an application conforming to this standard supporting at least:

   1) application international product;

   2) legacy product handling using the wrapper data group.

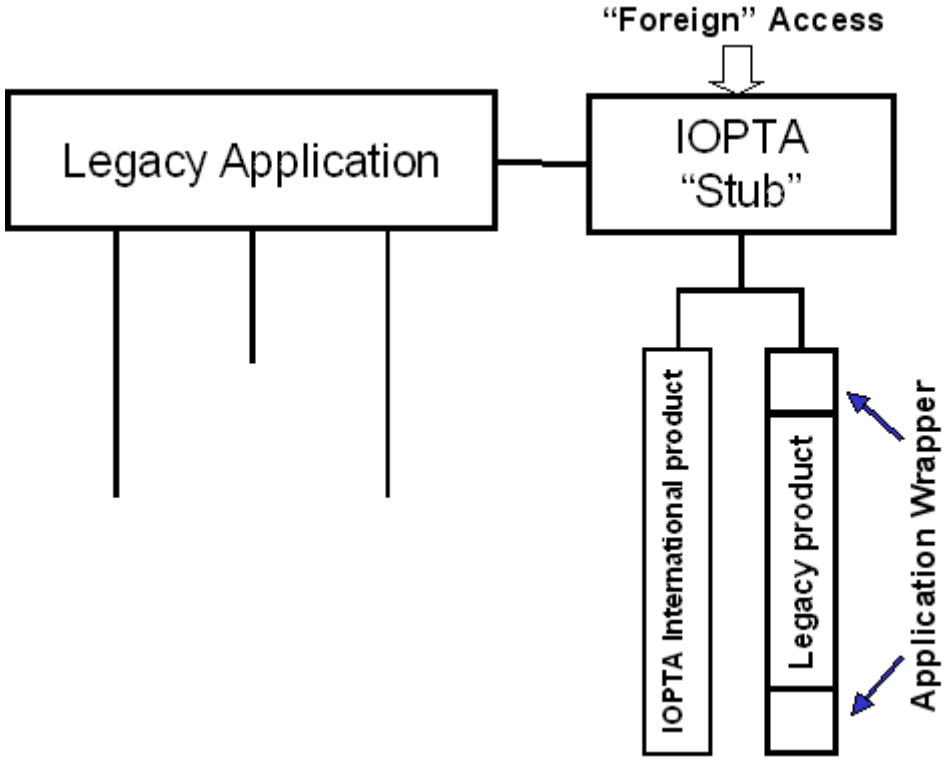

Figure I.1 — Interoperable Public Transport Compliant application

Bibliography

[1] ISO/IEC 7816-9, *Identification cards — Integrated circuit(s) card with contacts — Part 9: Commands for card management*

[2] ISO/IEC 8824 (all parts), *Information technology — Abstract Syntax Notation One (ASN.1)*

[3] ISO/IEC 8825 (all parts), *Information technology — ASN.1 encoding rules*

[4] ISO/IEC 14443-1, *Identification cards — Contactless integrated circuit(s) cards — Proximity cards — Part 1: Physical characteristics*

[5] ISO/IEC 14443-2, *Identification cards — Contactless integrated circuit(s) cards — Proximity cards — Part 2: Radio frequency power and signal interface*

[6] ISO/IEC 14443-3, *Identification cards — Contactless integrated circuit(s) cards — Proximity cards — Part 3: Initialisation and anticollision*

[7] ISO/IEC 14443-4 *Identification cards — Contactless integrated circuit(s) cards — Proximity cards — Part 4: Transmission protocol*

BSI — British Standards Institution

BSI is the independent national body responsible for preparing British Standards. It presents the UK view on standards in Europe and at the international level. It is incorporated by Royal Charter.

Revisions

British Standards are updated by amendment or revision. Users of British Standards should make sure that they possess the latest amendments or editions.

It is the constant aim of BSI to improve the quality of our products and services. We would be grateful if anyone finding an inaccuracy or ambiguity while using this British Standard would inform the Secretary of the technical committee responsible, the identity of which can be found on the inside front cover. Tel: +44 (0)20 8996 9000. Fax: +44 (0)20 8996 7400.

BSI offers members an individual updating service called PLUS which ensures that subscribers automatically receive the latest editions of standards.

Buying standards

Orders for all BSI, international and foreign standards publications should be addressed to Customer Services. Tel: +44 (0)20 8996 9001. Fax: +44 (0)20 8996 7001. Email: orders@bsi-global.com. Standards are also available from the BSI website at http://www.bsi-global.com.

In response to orders for international standards, it is BSI policy to supply the BSI implementation of those that have been published as British Standards, unless otherwise requested.

Information on standards

BSI provides a wide range of information on national, European and international standards through its Library and its Technical Help to Exporters Service. Various BSI electronic information services are also available which give details on all its products and services. Contact the Information Centre. Tel: +44 (0)20 8996 7111. Fax: +44 (0)20 8996 7048. Email: info@bsi-global.com.

Subscribing members of BSI are kept up to date with standards developments and receive substantial discounts on the purchase price of standards. For details of these and other benefits contact Membership Administration. Tel: +44 (0)20 8996 7002. Fax: +44 (0)20 8996 7001. Email: membership@bsi-global.com.

Information regarding online access to British Standards via British Standards Online can be found at http://www.bsi-global.com/bsonline.

Further information about BSI is available on the BSI website at http://www.bsi-global.com.

Copyright

BSI
389 Chiswick High Road
London
W4 4AL