



BSI Standards Publication

Identification card systems — European Citizen Card

Part 3: European Citizen Card
Interoperability using an application
interface

NO COPYING WITHOUT BSI PERMISSION EXCEPT AS PERMITTED BY COPYRIGHT LAW

raising standards worldwide™

Full document details: DD CEN/TS 15480-3:2010

National foreword

This Draft for Development is the UK implementation of CEN/TS 15480-3:2010.

This publication is not to be regarded as a British Standard.

It is being issued in the Draft for Development series of publications and is of a provisional nature. It should be applied on this provisional basis, so that information and experience of its practical application can be obtained.

Comments arising from the use of this Draft for Development are requested so that UK experience can be reported to the international organization responsible for its conversion to an international standard. A review of this publication will be initiated not later than 3 years after its publication by the international organization so that a decision can be taken on its status. Notification of the start of the review period will be made in an announcement in the appropriate issue of *Update Standards*.

According to the replies received by the end of the review period, the responsible BSI Committee will decide whether to support the conversion into an international Standard, to extend the life of the Technical Specification or to withdraw it. Comments should be sent to the Secretary of the responsible BSI Technical Committee at British Standards House, 389 Chiswick High Road, London W4 4AL.

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.

© BSI 2011

ISBN 978 0 580 69792 0

ICS 35.240.15

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

This Draft for Development was published under the authority of the Standards Policy and Strategy Committee on 31 January 2011.

Amendments issued since publication

Date	Text affected
------	---------------

TECHNICAL SPECIFICATION
SPÉCIFICATION TECHNIQUE
TECHNISCHE SPEZIFIKATION

CEN/TS 15480-3

December 2010

ICS 35.240.15

English Version

**Identification card systems - European Citizen Card - Part 3:
European Citizen Card Interoperability using an application
interface**

Systèmes d'Identification par Carte - Carte Européenne de
Citoyen - Partie 3: Interopérabilité de la Carte européenne
de Citoyen par interface applicative

Identifikationskartensysteme - Europäische Bürgerkarte -
Teil 3: Anwendungsschnittstelle für die Interoperabilität von
Europäischen Bürgerkarten

This Technical Specification (CEN/TS) was approved by CEN on 12 July 2010 for provisional application.

The period of validity of this CEN/TS is limited initially to three years. After two years the members of CEN will be requested to submit their comments, particularly on the question whether the CEN/TS can be converted into a European Standard.

CEN members are required to announce the existence of this CEN/TS in the same way as for an EN and to make the CEN/TS available promptly at national level in an appropriate form. It is permissible to keep conflicting national standards in force (in parallel to the CEN/TS) until the final decision about the possible conversion of the CEN/TS into an EN is reached.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Croatia, 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.



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

Management Centre: Avenue Marnix 17, B-1000 Brussels

© 2010 CEN All rights of exploitation in any form and by any means reserved
worldwide for CEN national Members.

Ref. No. CEN/TS 15480-3:2010: E

Contents

Page

Foreword.....	6
1 Scope	7
2 Normative references	7
3 Terms and definitions	8
4 Symbols and abbreviations	8
4.1 Abbreviations	8
5 ECC fitting in ISO/IEC 24727 model	11
5.1 ISO/IEC 24727 main features	11
5.2 General security issues – Applicable 24727-4 Stack Configurations for the ECC environment	13
5.3 ECC-3 Middleware Architecture	16
5.3.1 Service Access Layer (SAL)	17
5.3.2 Generic Card Access Layer (GCAL)	17
5.3.3 Interface Device Layer and API (IFD API)	17
5.3.4 ECC-3 Stack Distribution and Connection Handling	17
5.3.5 A Web Service based architecture for ECC-3 framework.....	21
5.3.6 XML-based SAL interface	26
5.3.7 Smart card profile fitting with ECC-3 stack.....	26
6 Card Discovery Mechanisms	27
6.1 Discovery decision tree	28
6.2 Migration path towards ECC and provision for legacy cards	29
6.2.1 Interoperable access to the Repository	30
6.3 Set of data for interoperability.....	32
6.4 Application and Card Capability Descriptors	32
6.5 ISO/IEC 7816-15 implementation.....	34
6.5.1 Profile designation within EF.DIR	35
6.5.2 ISO/IEC 24727-3 data structures mapping	35
6.5.3 SAL-API Action mapping onto ISO/IEC 7816-15 attributes	51
6.5.4 ISO/IEC 24727-3 data structures storage onto the card	53
6.5.5 General discovery mechanism.....	55
6.6 Other data descriptor	57
7 Authentication protocols	57
7.1 Authentication Mechanisms based on ISO/IEC 24727 SAL-API	57
7.2 Asymmetric internal authentication.....	58
7.3 Asymmetric external authentication.....	58
7.4 Symmetric internal authentication.....	58
7.5 Symmetric external authentication.....	59
7.6 Mutual authentication with key establishment.....	59
7.7 Device authentication with non traceability.....	59
7.8 Key transport protocol based on RSA	59
7.9 Terminal Authentication.....	60
8 IFD-API Web Service Binding	60
8.1 Specification of ISOCCommon.XSD	60
8.2 Specification of ISOIFD.XSD.....	61
8.3 Specification of CENIFD.WSDL	74
8.4 Specification of CENIFDCallback.XSD	83
8.5 Definition of CENCallback.WSDL.....	84

9	Card-Info Structure.....	85
9.1	Introduction.....	85
9.2	Overview.....	86
9.3	CardType	87
9.4	CardIdentification	88
9.5	CardCapabilities	94
9.6	ApplicationCapabilities.....	103
9.7	Signature	109
9.8	Complete XML-Schema Definition	109
10	XML-based Service Access Layer Interface	112
10.1	XML-Schema definitions for Service Access Layer functions	112
10.2	WSDL definitions for Service Access Layer functions	137
Annex A	(informative) Interface Device Layer Architecture and Management.....	161
A.1	Scope	161
A.2	IFD-Layer Architecture.....	161
A.3	Resource Manager	162
A.3.1	IFD-Handlers	162
A.3.2	Card transactions	162
A.3.3	Application threads	162
A.4	Administrative functions	162
A.4.1	IFD-Handler related functions	162
A.4.2	Interface Device related functions	163
A.5	IFD-Handler-API	163
Annex B	(informative) Interface Device API	164
B.1	Card terminal related functions	164
B.1.1	EstablishContext	164
B.1.2	ReleaseContext.....	165
B.1.3	ListIFDs.....	165
B.1.4	GetIFDCapabilities.....	166
B.1.5	GetStatus.....	168
B.1.6	Wait	170
B.1.7	Cancel.....	171
B.1.8	ControlIFD	172
B.2	Card related functions	172
B.2.1	Connect	173
B.2.2	Disconnect	174
B.2.3	BeginTransaction	174
B.2.4	EndTransaction	175
B.2.5	Transmit.....	175
B.3	User related functions.....	176
B.3.1	VerifyUser.....	177
B.3.2	ModifyVerificationData.....	179
B.3.3	Output.....	181
Annex C	(informative) IFD-API – C Language Binding.....	183
Annex D	(informative) Examples of Cryptographic Information Application for Card-Application	
	Service Description	189
D.1	Fetching a certificate for internal asymmetric authentication.....	189
D.2	Creating a new service.....	190
D.2.1	Features of eVoting Service	190
Annex E	(informative) SAL-API Post-issuance personalization requests	204
E.1	Post-issuance personalization requests.....	204
E.2	Canonical protocol	204
E.2.1	DataSetCreate	205
E.2.2	DSICreate.....	213
E.2.3	DIDCreate	214
E.2.4	DIDUpdate	216

E.2.5	CardApplicationServiceCreate	216
Annex F	(informative) Additional features versus ISO/IEC 24727	219
F.1	Discovery Mechanism	219
F.2	General Procedures (SAL)	220
F.3	Architecture	221
F.4	eURI support (through ControllFD() call)	222
F.5	Differences between IFD-API in ISO/IEC 24727-4 and ECC-3	222
F.5.1	More generale SlotCapabilityType	222
F.5.2	Transmit with support for batch processing	222
F.5.3	Additional error code for Signalevent	222
F.6	Miscellaneous corrections	222
Annex G	(informative) C-Language Binding for ExecuteSAL function	223
Annex H	(informative) Java-Language Binding for ExecuteSAL function	224
Annex I	(informative) XML-Binding for Authentication Protocols	225
I.1	PIN Compare	225
I.1.1	Marker	225
I.1.2	DIDCreate	232
I.2	Mutual authentication	234
I.2.1	Marker	235
I.3	RSA Authentication	240
I.3.1	Marker	241
I.3.2	DIDCreate	244
I.3.3	DIDUpdate	244
I.3.4	DIDGet	244
I.3.5	CardApplicationStartSession	244
I.3.6	DIDAuthenticate	245
I.4	Generic cryptography	248
I.4.1	Marker	249
I.4.2	DIDCreate	254
I.4.3	DIDUpdate	254
I.4.4	DIDGet	254
I.4.5	Encipher	254
I.4.6	Decipher	254
I.4.7	GetRandom	254
I.4.8	Hash	254
I.4.9	Sign	254
I.4.10	VerifySignature	254
I.4.11	VerifyCertificate	254
I.4.12	DIDAuthenticate	255
Annex J	(informative) API for ISO/IEC 7816-15 data structures handling	257
J.1	C-language Binding for the ECC3-API	259
J.1.1	ECC3RESULT	259
J.1.2	ECC3CONTEXT	259
J.1.3	ECC3INFO	259
J.1.4	ECC3VERSION	260
J.1.5	CioChoice	260
J.1.6	CommonObjectFlags	260
J.1.7	SecurityEnvironmentInfo	260
J.1.8	AlgorithmInfo	261
J.1.9	PasswordType	261
J.1.10	Validity	261
J.1.11	ObjectValueType	261
J.1.12	FileType	262
J.1.13	FileState	262
J.1.14	IdType	262
J.1.15	AccessModes	263
J.1.16	Operations	263

J.1.17	ContextTag	263
J.1.18	SecurityConditionType	264
J.1.19	DataSetNameType	264
J.1.20	DSINameType	264
J.2	Interface functions	265
J.2.1	General Purposes Functions.....	265
J.2.2	Reader and Card management Functions	265
J.3	Objects.....	266
J.3.1	Basic objects	266
J.3.2	File Objects	275
J.3.3	Data Objects.....	283
J.4	Macros	292
J.4.1	_HB: HexaBlob conversions	292
J.4.2	AsString.....	293
J.5	Example of use (C++ Language).....	293
Annex K	(informative) Global Profile 4: card requirements to access/offer services in ISO/IEC 24727 framework	295
K.1	Global Profile 4: Card requirements	295
K.1.1	OID	295
K.1.2	General	295
K.1.3	interfaces / transport protocols	295
K.1.4	Data elements and data structures.....	296
K.1.5	Command set.....	298
K.1.6	Data structure of Card Applications	299
Bibliography		300

Foreword

This document (CEN/TS 15480-3:2010) 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.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN [and/or CENELEC] shall not be held responsible for identifying any or all such patent rights.

According to the CEN/CENELEC Internal Regulations, the national standards organizations of the following countries are bound to announce this Technical Specification: Austria, Belgium, Bulgaria, Croatia, 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 the United Kingdom.

1 Scope

ECC part 3 will provide an Interoperability Model, which will enable an eService compliant with technical requirements, to interoperate with different implementations of the European Citizen Card.

This Interoperability model will be developed as follows:

- starting from the ECC part 2, part 3 of the ECC series will provide additional technical specifications for a middleware architecture based on ISO/IEC 24727. This middleware will provide an API to an eService as per ISO/IEC 24727-3;
- a set of additional API provide the middleware stack with means to facilitate ECC services;
- a standard mechanism for the validation of the e-ID credential stored in the ECC and retrieved by the service.

In order to support the ECC services over an ISO/IEC 24727 middleware configuration, this part of the standard specifies the following:

- a set of mandatory requests to be supported by the middleware implementation based on ISO/IEC 24727;
- data set content for interoperability to be personalized in the ECC;
- two middleware architecture solutions: one based on a stack of combined ISO/IEC 24727 configurations and the other based on Web Service configuration;
- a Global Profile featuring the guidelines for card-applications to fit in ISO/IEC 24727 framework.

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 14890-1:2008, *Application Interface for smart cards used as Secure Signature Creation Devices — Part 1: Basic services*

ISO/IEC 7816-3:2008, *Identification cards — Integrated circuit cards — Part 3: Cards with contacts — Electrical interface and transmission protocols*

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

ISO/IEC 7816-8:2004, *Identification cards — Integrated circuit cards — Part 8: Commands for security operations*

ISO/IEC 7816-9, *Identification cards — Integrated circuit cards — Part 9: Commands for card management*

ISO/IEC 7816-15:2004, *Identification cards — Integrated circuit cards with contacts — Part 15: Cryptographic information application*

ISO/IEC 7816-15:2004/Amd 2:2008, *Identification cards — Integrated circuit cards — Part 15: Cryptographic information application, Error corrections and extensions for multi-application environments*

ISO/IEC 9796-2:2002, *Information technology — Security techniques — Digital signature schemes giving message recovery — Part 2: Integer factorization based mechanisms*

ISO/IEC 24727-1:2007, *Identification cards — Integrated circuit card programming interfaces — Part 1: Architecture*

ISO/IEC 24727-2:2008 *Identification Cards — Integrated circuit card programming interfaces — Part 2: Generic card interface*

ISO/IEC 24727-3:2008 *Identification Cards — Integrated circuit card programming interfaces — Part 3: Application Interface*

ISO/IEC 24727-4:2008, *Identification cards — Integrated circuit card programming interface — Part 4: Application programming interface (API) Administration*

3 Terms and definitions

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

3.1 descriptive elements

the Application Capability Descriptor (ACD) and the Card Capability Descriptor (CCD) comprehend information nested in data objects and intended for the discovery mechanism. These descriptive elements are encapsulated along with procedural elements in the ACD and CCD. EF.DIR is part of the descriptive elements

3.2 procedural elements

translation code to process any request at the Generic Card Interface (GCI) and every relevant card response. The translation has one entry point, theTranslationCode() function as per ISO/IEC 24727-2

3.3 middleware

set of abstraction layers that serves as the intermediate between a client-application and an application resident in the ECC. The actual pieces of software running behind these abstraction layers are implementation-specific and out of the scope of this document

3.4 service

an eService is an application based locally on the client PC or based somewhere in the internet (e.g. government eService, eBuisness eService...) which offers in combination with the ECC smart card the execution of a task

4 Symbols and abbreviations

4.1 Abbreviations

ADF	Application Dedicated File
AID	Application Identifier
AJAX	Asynchronous JavaScript and XML
AMB	Access Mode Byte
AT	Authentication Template

ATR	Answer to Reset
ATS	Answer to Select
BER	Basic Encoding Rules
BHT	Biometric Header Template
BIT	Biometric Information Template
CA	Certification Authority
CAPICOM	Cryptographic API COM Object (http://msdn.microsoft.com)
CAR	Certification Authority Reference
CBC	Cipher Block Chaining
CCT	Cryptographic Checksum Template
CED	Certificate Effective Date
CHA	Certificate Holder Authorisation
CHR	Certificate Holder Reference
CIA	Cryptographic Information Application
CPI	Certificate Profile Identifier
CRT	Control Reference Template
CryptoAPI	Cryptographic Application Programming Interface (http://msdn.microsoft.com)
CSP	Cryptographic Service Provider
CT	Confidentiality Template
CV	Card Verifiable
CXD	Certificate Expiration Date
DES	Data Encryption Standard
DF	Dedicated File
DH	Diffie Hellman
DOCP	Data Object Control Parameters
DST	Digital Signature Template
ECDH	Elliptic Curve DH
ELC	Elliptic Curve Cryptosystem
ECDSA	Elliptic Curve Digital Signature Algorithm

EF	Elementary File
FCI	File Control Information
FCP	File Control Parameters
GCAL	Generic Card Access Layer
HT	Hash Template
ICC	Integrated Circuit Card
DID	Differential Identity acc. ISO/IEC 24727-3
IFD	Interface Device
IFDH	Interface Device Handler
JSON	Java Script Object Notation (http://www.json.org)
KAT	Control reference template for key agreement
LSB	Least Significant Byte
MAC	Message Authentication Code
MF	Master File
M.U.S.C.L.E	Movement for the Use of Smart Card in Linux Environment
MSE	Manage Security Environment
OID	Object Identifier
PAN	Primary Account Number
PIN	Personal Identification Number
PUK	unblocking password
PK – DH	Public key – Diffie Hellman (asymmetric key base algorithm)
PSO	Perform Security Operation
RFU	Reserved for Future Use
RSA	Rivest Shamir Adleman
SAL	Service Access Layer
SDO	Security Data Object
SCB	Security Condition Byte
SE	Security Environment
SEID	Security Environment IDentifier byte

SM	Secure Messaging
SSD	Security Service Descriptor
TLV	Tag Length Value
UQB	Usage Qualifier Byte

5 ECC fitting in ISO/IEC 24727 model

5.1 ISO/IEC 24727 main features

This standardization initiative (ISO/JTC1/SC17 WG4/TF9) aims to design a new framework for interoperability based on a discovery mechanism of which the following paradigm:

The low level implementation features of the smart card, including proprietary specific features and characteristics based on ISO standards (i.e. ISO/IEC 7816-4) are hidden to the client-application through a high-level description provided to the terminal and processed by the middleware stack.

The middleware stack is defined by abstraction layers ensuring the interoperability. These layers are sustained by applicative components of which the implementation is up to the integrator and may vary according to the environment. For instance, the use of CAPICOM, CryptoAPI, CSP, proprietary API or DLL, ActiveX objects, Applets, PKCS#11 interface, JCE, M.U.S.C.L.E PC/SC emulation on Linux environment etc., are all possible solutions upon which the ISO/IEC 24727 abstraction layers may run.

The middleware abstraction layers are of two kinds:

- the Service Access Layer (SAL) is in charge of interpreting the requests addressed by the client-application to the card via a high-level API (the SAL-API). The SAL translates the requests in terms of sequences of APDU that are sent out to the underlying abstraction layer (the GCAL). This translation is performed according to the rules defined by a set of interoperability data reflecting the rules governing the card-applications. The SAL shall generate on the fly these interoperability data out of the ISO/IEC 7816-15 information available in the card. This ISO/IEC 7816-15 information is either provided within the CardApplicationServiceDescription data, or within the DF.CIA files, or both. Upon request from the eService, the SAL may surface the interoperability data to the eService. The SAL is specified in ISO/IEC 24727-3;
- the Generic Card Access Layer (GCAL) is in charge of translating the APDU handed on by the SAL in terms of APDU understandable to the smart card. This translation is applied according to the rules defined in the ACD (Application Capabilities Descriptor) and/or in the CCD (Card Capabilities Descriptor) templates. These templates are read out of the card by the GCAL. The GCAL performs a bootstrap mechanism upon card detection in order to retrieve the ACD and CCD containers from the card. The bootstrap operation is the first step of the discovery mechanism. The GCAL functionality is specified in ISO/IEC 24727-2.

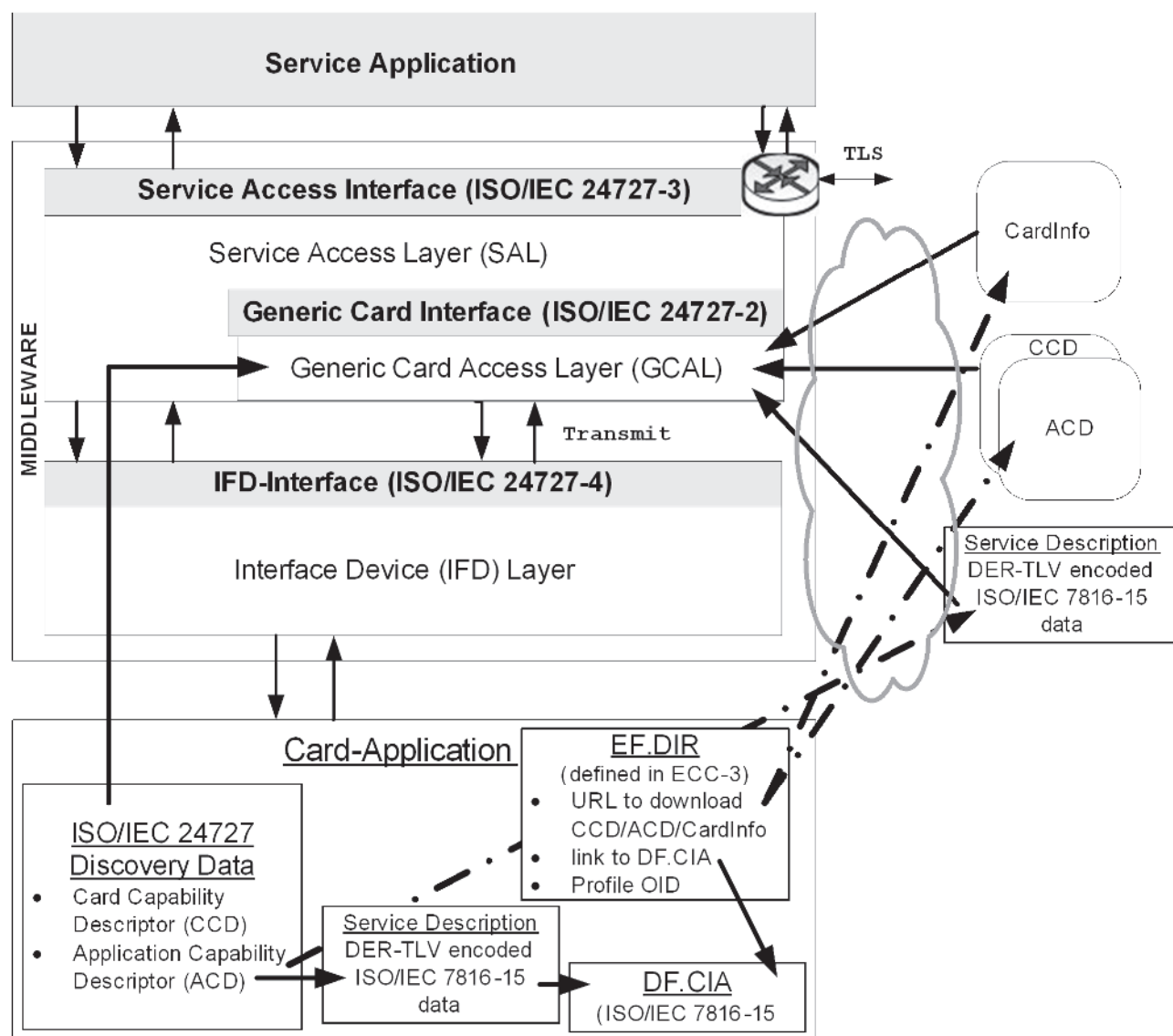


Figure 1 — CEN/TS 15480 compliant smart card in ISO/IEC 24727 framework

The smart card hosts a set of interoperability data that are DER-TLV encoded according to an ASN.1 definition. This ASN.1 specification provided in ISO/IEC 24727-2 describes all the information items required by a terminal to access the services offered by the card (resources, actions) and to allow as well the end-user to access services running on terminal side. This ASN.1 specification is built upon the computational model specified in ISO/IEC 24727-3. Therefore, the ASN.1 definition encompasses the descriptors of the services hosted on-card per application, the card application list, the access control rules applying to each on-card action or resource, and the description of each modular authentication item (called Differential-Identity or DID). This latter item (Figure 2 — Differential-Identity structure according to ISO/IEC 24727-3) is the modular descriptor of each authentication procedure. DID is comprised of five fields among which an optional element, the *dIDQualifier*, that conveys further information to clear ambiguities whenever required by cryptographic requests.

Element 1	Element 2	Element 3	Element 4	Element 5
DIDName	Authentication Protocol	Marker	Scope (Global or Local; implicit)	dIDQualifier
Mandatory	Mandatory	Mandatory		Optional

Figure 2 — Differential-Identity structure acc. ISO/IEC 24727-3

The relationship between the elements constituting a DID is such as a Marker applied in a given Scope feeds the algorithm of a given Authentication Protocol that is applied to authenticate a given named DID.

The rules controlling the access to a card resource or authorizing a card action are expressed has a Boolean combination of Differential-Identities. If the Boolean expression evaluates to "TRUE", the requested card resources are made available or the card action is executed, otherwise the request is rejected.

The DifferentialIdentityQualifier may serve to convey to the SAL the OID related to a cryptographic operation or authentication protocol. The DIDQualifier type depends on the Authentication Protocol to which it applies. Details of DIDQualifiers are given in ISO/IEC 24727-3:2008, Annex A.

5.2 General security issues – Applicable 24727-4 Stack Configurations for the ECC environment

The abstraction layers defined by ISO/IEC 24727 are intended to hide seamlessly the implementation differences that may occur between smart cards from different vendors even when their respective implementation is based on the same standard or specification (i.e. CEN/TS 15480-2)

A high level description (according to ISO/IEC 24727) of the security rules governing the services hosted by the card allows and extension of the information provided by ISO/IEC 7816-15 implementation, ensuring thereby a more comprehensive interoperability protocol based on ISO/IEC 7816-15.

By personalizing the necessary data sets for interoperability as per ISO/IEC 24727-2 and ISO/IEC 24727-3, a European Citizen Card can fit in the ISO/IEC 24727 framework with the following limitations:

- the GCAL being assumed to translate APDU handed on by the SAL, these command APDU can not be secured (integrity, confidentiality) otherwise any change in the command header according to the ACD procedural elements will cause a rejection of the command by the smart card. Therefore, if the translation function of the GCAL is performed, end-to-end Secure Messaging cannot be applied between the client-application and the card-application in the ISO/IEC 24727 framework. With this respect, the European Citizen Card needs to discard the GCAL translation function unless for use cases where no Secure Messaging is envisioned;
- the Loyal-stack configuration (Figure 3 — ISO/IEC 24727-4 excerpt: Loyal-Stack configuration), the Remote-ICC-stack configuration (Figure 4 — ISO/IEC 24727-4 excerpt: Remote-ICC-Stack configuration), and the Remote-Loyal-Stack configuration (Figure 5 — ISO/IEC 24727-4 excerpt: Remote-Loyal-Stack configuration) are considered in the present document. A generic depiction of ISO 24727 middleware stack lying over different platforms is represented on Figure 6 — ISO/IEC 24727-4 excerpt: Generic elements of ISO/IEC 24727 MW stack);

- the implementation of Part 3 Layer (SAL) and Part 2 Layer (GCAL) on separate machines raises the communication channel security issue to bridge the two layers. The use of a TLS session in between to prevent eavesdropping, tampering or message forgery, raises further constraints:
 - for the client-application to keep control on the intermediate TLS secured communication channels it requires a handle on the TLS secure session and it needs to share the secrets involved in the secure session establishment. Such control is not provided by ISO/IEC 24727-4, nevertheless a parameter denoting the available routes between the client-application and the card-application is backed up to the client-application for awareness then path selection or securisation. During the first phase of the SSL session, the client and server negotiate which cryptographic algorithms will be used. In typical use, only the server is authenticated while the client remains unauthenticated. For the purposes of mutual authentication, a public key infrastructure (PKI) deployment to clients is required;
 - ISO/IEC 24727-3 and ISO/IEC 24727-4 do not provide to the client-application control over the secure messaging features and therefore require session keys sharing, which raises different security issues. It is preferable for the session keys and more generally for the host security module functions and keys to remain the property of the eService;
 - SSL runs on layers beneath HTTP, SMTP and NNTP application protocols, and above TCP or UDP transport protocol. SSL can be exploited to add security to any protocol using reliable connections like TCP. Therefore, to avoid using separate ports for encrypted communications and to enable the application protocols to upgrade to TLS from a plaintext connection, the client should support SSL natively instead of relying on standalone SSL products;
- the implementation of client-application and Part 3 Layer (SAL) on separate machines entails securing the requests between the SAL Proxy and the SAL Agent on both sides, and requires the marshallized requests to be transported over TLS. The marshalling protocol may be based on DER-TLV encoding according to ASN.1 definition, or on WSDL transported over SOAP. Both methods are described in ISO/IEC 24727-4.

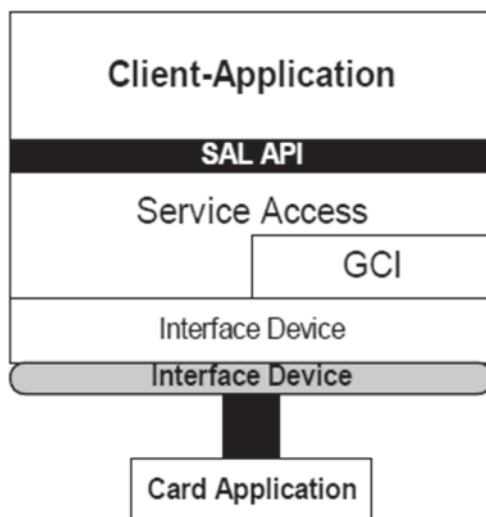


Figure 3 — ISO/IEC 24727-4 excerpt: Loyal-Stack configuration

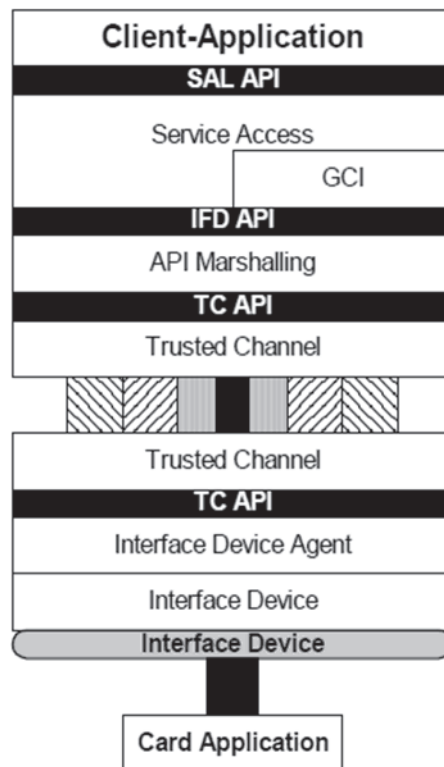


Figure 4 — ISO/IEC 24727-4 excerpt: Remote-ICC-Stack configuration

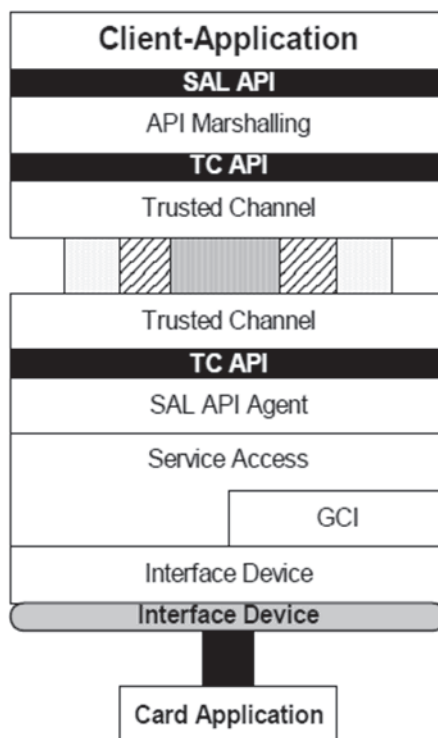


Figure 5 — ISO/IEC 24727-4 excerpt: Remote-Loyal-Stack configuration

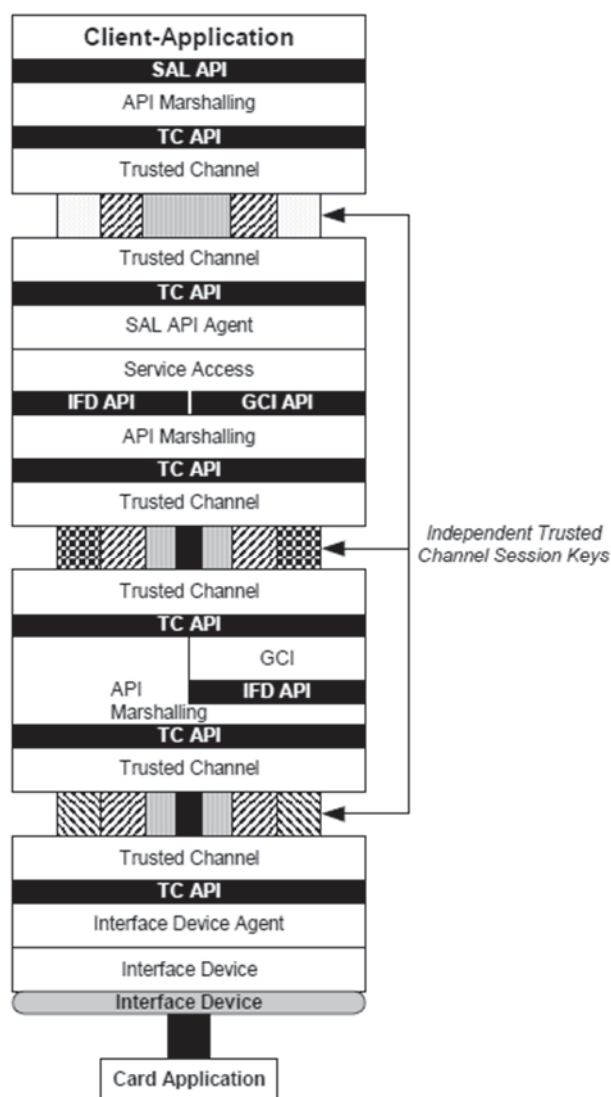


Figure 6 — ISO/IEC 24727-4 excerpt: Generic elements of ISO/IEC 24727 MW stack

5.3 ECC-3 Middleware Architecture

On overview of the architectural parts of ECC-3 is given in Figure 7 — ECC-3 Middleware Architecture. The ECC-3 Middleware consists of the Service Access Layer (SAL), the Generic Card Access Layer (GCAL) and the Interface Device Layer (IFD-API). A detailed description of the different parts of the ECC-3 Middleware can be found in the subsequent sections.

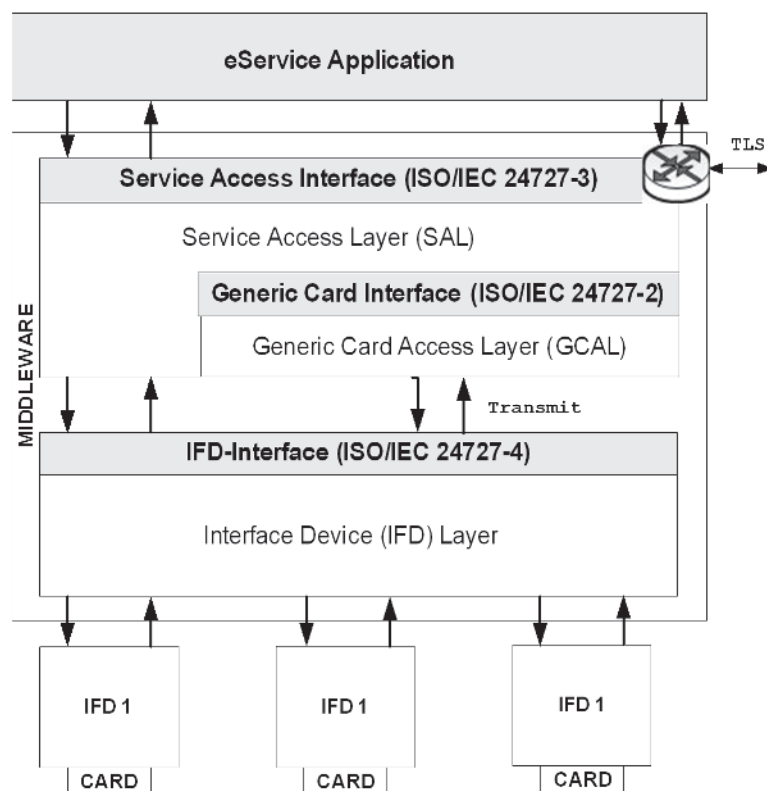


Figure 7 — ECC-3 Middleware Architecture

5.3.1 Service Access Layer (SAL)

The ECC-3 Service Access Layer is based on the Service Access Layer of ISO/IEC 24727-3.

5.3.2 Generic Card Access Layer (GCAL)

The Generic Card Access Layer is responsible for mapping the SAL functionality to card application specific functions and protocols. To allow an easy adaptation within the GCAL, the GCAL uses ACD, CCD and CardInfo files. Details can be found in 6.3.

5.3.3 Interface Device Layer and API (IFD API)

The Interface Device API is responsible for the APDU based communication with the IFD / card application.

5.3.4 ECC-3 Stack Distribution and Connection Handling

If the eService is located on a server somewhere on the network and the ECC is used by a user at home, the Remote-ICC-Stack, the Remote-Loyal-Stack, further to the Loyal-Stack configuration of ISO/IEC 24727-4, will be used within the ECC-3 middleware architecture. Within this distributed system, it is necessary that the eService will communicate directly with the card application. Additionally, the local client embedded application may perform an authentication process with the card application or may interact by other means with the card application. APDUs generated at service side in a Remote-ICC-Stack like layout may be addressed directly to the card through the IFD-API, and marshallized requests generated at server side in a Remote-Loyal-Stack like layout may be addressed to the SAL implementation on client machine. This implies, that the complete ECC-3 middleware stack must be available on the client machine and on the server side (see Figure 8 — ECC-3 Middleware Architecture in the Remote-ICC-Stack environment), and that a mix of different ISO/IEC 24727 configurations must be deployed together to achieve electronic service delivery in the real life.

This Technical Specification aims at providing guidelines for legacy card support to facilitate migration towards ECC-compliant cards. In this view, the ICC-Resident-Stack (ISO/IEC 24727 subclause 5.5) is a particular configuration such as the service layer processing the high-level calls from the eService application is embedded onto the card. The card-application emulating stack has an interface APDU exposing the ENVELOPE command. The eService application calls the SAL-API requests; the SAL-API requests are marshalled by a proxy according to a transmission protocol based on ASN.1 (ASN.1-Binding) as follows: the SAL-API calls are encoded in DER-TLV. The DER-TLV payload is encapsulated within the data field of the ENVELOPE APDU command and sent out to the card. Alternatively, the marshalling protocol may use XML-Binding.

Upon reception of the ENVELOPE command, the applet parses its data field through a DER-TLV encoder/decoder function or alternatively an XML-parser function. The result of the parsing is processed by the card-application according to ISO/IEC 24727 business logic.

One outstanding consequence of this implementation is that the card-application hosts directly the data structures employed by the ISO/IEC 24727 business logic. Usually for the rest of 24727 configurations (except ICC-Resident-stack), a discovery mechanism takes place to translate the card-application services and related security rules in terms of logical data structures (DataSet, DSI, DID, ACL, CardApplicationList, CardApplicationServiceList etc.): this *discovery mechanism* allows the middleware service layer (off-card) to discover the card function through a specific service-oriented language (based on ISO/IEC 7816-15). At initialization time, the card delivers a container to the middleware service layer. This bootstrap procedure provides the middleware with the ACD and CCD containers. The middleware service layer gets and parses this contents and generates out of it the logical data structures according to ISO/IEC 24727 business logic. Only these resulting data structures for interoperability are exposed to the eService application upon request. The eService application does not need to handle APDU (the ENVELOPE command being generated by the eService-Proxy layer).

The ICC-Resident-Stack is embedding the middleware service layer and therefore does not require ACD anymore. Only a CCD informing the middleware that the card conforms to the ICC-Resident-stack shall be available on-card. Consequently, this configuration spares the encoding and personalization of ACD on the card.

This configuration has the advantage of facilitating the post-issuance creation/personalization of data structures onto the card: the ISO/IEC 24727 data structures are directly hosted on-card as DER-TLV encoded Data Object. Personalizing ISO/IEC 24727-compliant cards in post-issuance raises the issue of formatting properly the structures to be created, i.e. a DataSet shall be interpreted as a DF, EF or DO; a DSI shall be interpreted as a EF or DO; the EF shall be of a type (record, transparent, TLV etc) that has to be determined according to the card features; those translations that are likely to make post-issuance personalization complicate in common cases (see Annex E) are dispensed with for ICC-Resident-Stack.

The drawback of this configuration is mainly the lack of support for legacy: while existing cards in the field can become 24727-enabled by simple supply of their respective ACD/CCD, the ICC-Resident-stack embeds a newly designed/issued card-application and may be convenient for newly issued cards, without need for legacy requirement.

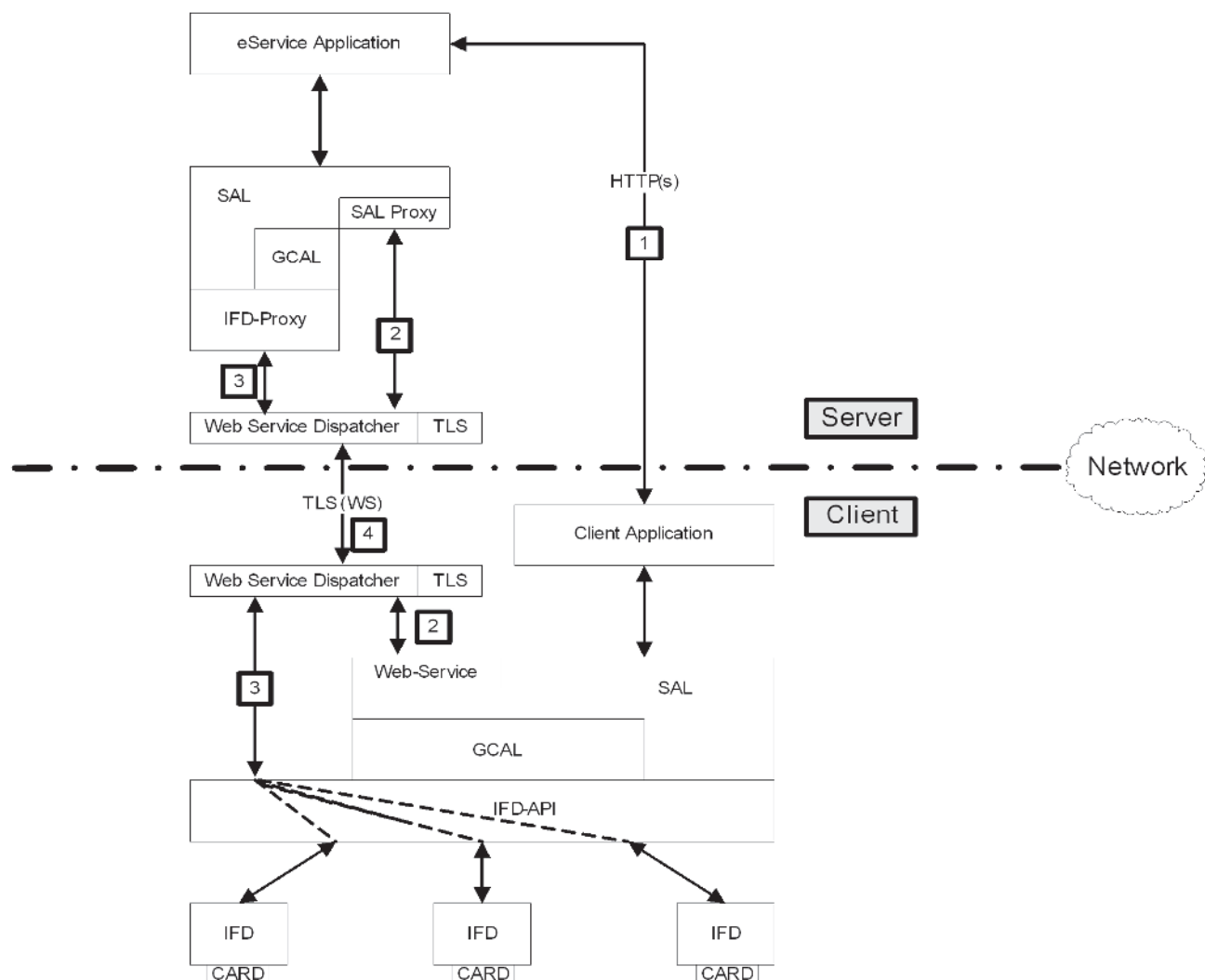


Figure 8 — ECC-3 Middleware Architecture in the Remote-ICC-Stack environment

If the user wants to use an eService, the communication will be started by opening a secure channel from the client application to the eService **1**. To allow a communication based on high level commands (SAL), the connection **2** will be established. Connection **3** is responsible for the APDU based communication between the eService and the card application. This connection allows the end-to-end authentication and secure channel between the card application and the eService. Finally, connection **4** will be used to transport the web service requests and responses between the client and the server.

The three different connections are necessary, since the communication end points for each communication channel are different.

Since connection **1** is application specific, it is out of scope for this specification. The connection handling for connection **2**, **3** and **4** are described in here after. Due to the usual firewall restrictions, these connection handling methods are designed such as the client will start the connection establishment.

5.3.4.1 Connection handling for the distributed stack components

As the ECC specification uses web service based communication instead of the trusted channel API as defined in ISO/IEC 24727-4 it is necessary to specify an analogous XML-based binding for the SAL functions. This binding addresses the *connection establishment between distributed systems*.

Some of the protocols specified here involve two SAL-instances, which run on different systems (eService and Client) and communicate with each other via potentially insecure networks. Therefore, the relevant aspects of security have to be taken into consideration when setting up the transport channel.

5.3.4.1.1 General security requirements

The security requirements of ISO/IEC 24727 dictate that the TLS protocol in accordance with RFC 4346 **MUST** be used.

Moreover, public server services **MUST** have access to corresponding X.509 certificates.

When it comes to the security of the communication between the different modules realizing the ECC-3 stack consisting of Service Access Layer (SAL) and IFD Layer), however, X.509 certificates **MAY** be used, whereby the associated private keys are to be adequately protected. Alternatively, anonymous TLS cipher suites, such as `TLS_DH_anon` from RFC 4346 or `TLS_ECDH_anon` from RFC 4492, **MAY** be used, although appropriate security measures need to be taken in the operational environment in order to avert man-in-the-middle attacks while the connection is being established.

In both cases there **MUST** be an exclusive binding of the communication context at application level to the TLS channel which has been established in this process. This communication context is established on connection to the IFD-Layer via the function `EstablishContext` and represented by the `ContextHandle`. When connecting to the SAL, this communication context corresponds to a connection to the card application established by means of `CardApplicationConnect`, which is represented by a `ConnectionHandle`.

As such, one single TLS channel is typically sufficient to establish communication between a SAL and the IFD layer — irrespective of the number of card terminals and connected cards — whereas a separate TLS channel is required for every connection to a card application for communication to take place between the identity layer or the application layer and the SAL.

5.3.4.1.2 Connections for SOAP binding

When using the SOAP binding SOAP-v1.1, the connection is established simply by setting up a TLS-protected channel between the user of the web service (service consumer) and the provider of the web service (service provider) via which web service messages can henceforth be exchanged. In this case, the service consumer and service provider take the roles of TLS/http client and TLS/http server, respectively.

5.3.4.1.3 Connections for PAOS binding

When using the PAOS binding PAOS-v1.1, however, a more complex process is required to establish the connection as, in this case, the TLS/http server acts as the user of the web service (service consumer) with eService because the TLS/http client acts as the provider of the web service (service provider) and **MUST** initiate the connection.

Moreover, in this case there are typically two different TLS channels, and appropriate cryptographic mechanisms **MUST** be used to safeguard their logical relationship while the connection is established.

The general connection sequence is shown in Figure 9.

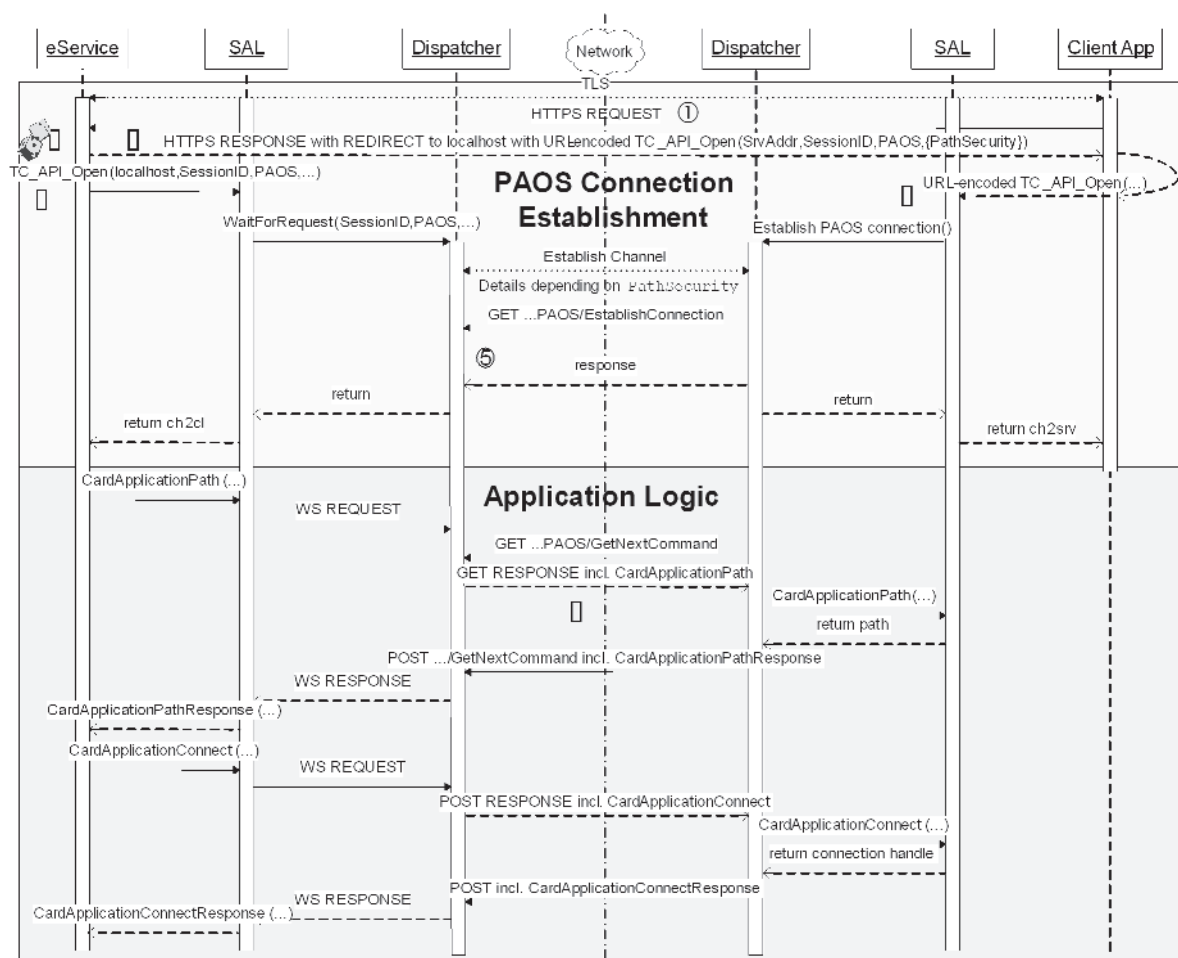


Figure 9 — General connection process with PAOS binding

5.3.5 A Web Service based architecture for ECC-3 framework

This section describes, as an alternative to subclause 5.3.4, a smartcard connectivity solution based on open standards (XML, CSS, HTML, HTTP, AJAX, JavaScript) and enabling web services (eAdmin or other) to communicate with a ECC through the widely available PC/SC layer that is part of the host operating system. The IFD layer (Annex A, A.2) elaborated in the present standard encapsulates the PC/SC layer.

The ISO/IEC 24727-4 Remote-ICC-Stack configuration is appropriately achieved by this solution.

5.3.5.1 Connectivity principles

The end user accesses a web service via his browser. This service execution is reflected through web pages embedding JavaScript that is generated by the Server and that may communicate with the smartcard. Either directly built in the web browser or indirectly via plug-in, the web page's script communicates with the hardware on the end user's machine (host). The default functionality of the browser may be extended to create a channel to the IFD stack (alternatively PC/SC stack) in the host machine. This extension is accessible via JavaScript and may be adapted to each type of browser (Internet Explorer, FireFox and others). Browsers being different in the way they support extensions, a library script (.js file) shall be made available by the Server to hide all the browser-dependent code from the eService Application. This library may provide as well an object oriented interface to the IFD layer.

The APDU commands are generated on server side and by the SAL-Implementation. The SAL-implementation calls the IFD-API and the resulting IFD-API requests are serialized according the WSDL binding described in Clause 8. The SOAP transmission payload is conveyed over HTTP and resolved on the end user's browser by the means of embedded JavaScript. The IFD-API call is unmarshalled and the APDU commands are extracted from the payload and handled by the embedded script that relies on the local library script extension. This library script wraps the IFD layer. The APDU responses from the ECC are managed the other way around and returned to the SOAP server.

For security purposes, before to grant access to its resources, the smartcard application may authenticate the Server. Secure APDUs may be conveyed as part of the SOAP payload across the network between the web service application and the end user browser as described above. ECC authentication mechanisms can take place in this framework without restriction.

The code that executes in the client (i.e. javascript) makes call to the server side code to either get more information from it or to submit information to it. This call is typically done by XMLHttpRequest object which is built into all the browsers. This object facilitates the connectivity between the javascript and the server in a browser/server platform/application agnostic way. Some code executing in javascript tells this object to transmit data to a server endpoint. The XMLHttpRequest is capable of transmitting and receiving not only XML-based data but any text generated from javascript. In case of processing overhead for the code (javascript) that executes in the browser (i.e. preliminary XML parsing and creation of an object tree for incoming data), XML payload may be replaced with the Javascript Native Object Notation (JSON) offering similar capabilities as that of XML. Therefore, JSON may alternatively be used to exchange data between javascript and the server side code with the following benefits:

- programming language-independent;
- human readable;
- presenting a structured view of data;
- less verbose than XML whereby saving the bandwidth;
- transformable syntax to an object programming language without need for parsing.

NOTE In case of use, JSON shall be mapped onto the XML definitions provided in this Technical Specification.

Table 1 shows the main software components and related layout realizing the connectivity solution described in this clause.

Table 1 — Remote-ICC-Stack configuration as a web service

Software components							
ISO/IEC 24727-4 Remote-ICC-Stack			End-user's machine	Web server		example of implementation	
1	Client-application			M		Web service running on application server Tomcat	
2	SAL-API			M		Acc. ISO/IEC 24727-3	
3	SAL-Implementation Layer			M		Acc. ECC-3	
4	Generic Card Interface	IFD-API		O	M	Based on Annex B	
5	IFD-Proxy			M		Marshalling acc. chapter 8	
6	WSDL Binding: Based on SOAP protocol over HTTP ¹			M		Web app. server Apache SOAP.	
						JavaMail package (mail.jar)	SOAP package (soap.jar, soap.war)
						Java Activation Framework package (activation.jar)	Xerces package (XML parser) (xerces.jar)
						Application server Tomcat	
						Java VM	
7	SOAP request/response over http(s)					SSL session	
8	BROWSER		M			(IE, Firefox or other)	
9	Connectivity solution	IFD-Agent : XML parser script acting as a <Marshaller.js>	M			Embedded JavaScript, java applet or/and JSP, ActiveX	
10		script accessing smart card (.JS script): PC/SC based	M			Embedded JavaScript	
11		Library script wrapping the IFD layer	M			Plug-in loaded securely from the web server	

¹ Alternatively, XML may be translated into JSON.

Table 1 (*continued*)

Software components							
ISO/IEC 24727-4 Remote-ICC-Stack		End-user's machine		Web server	example of implementation		
12	IFD-Layer		M			Annex A.2	
13	Smart Card Resource Manager		M				
14	Reader handler (driver)		M			PC/SC compatible IFDH	
15	Reader	Enhanced reader	M	O			IFD-API is recommended for enhanced readers
16	ECC		M			ECC with global profile {OID = 1.3.162.15480.4.4}	
17	ACD/CCD hosted on-card for the discovery mechanism or loaded form remote for legacy support		O			CardApplicationService-Description value acc. ECC-3 (see Annex D eVoting example) : DER-TLV encoded information representing the card-application resources for a given eService.	

5.3.5.2 Web service ECC-3 Middleware architecture

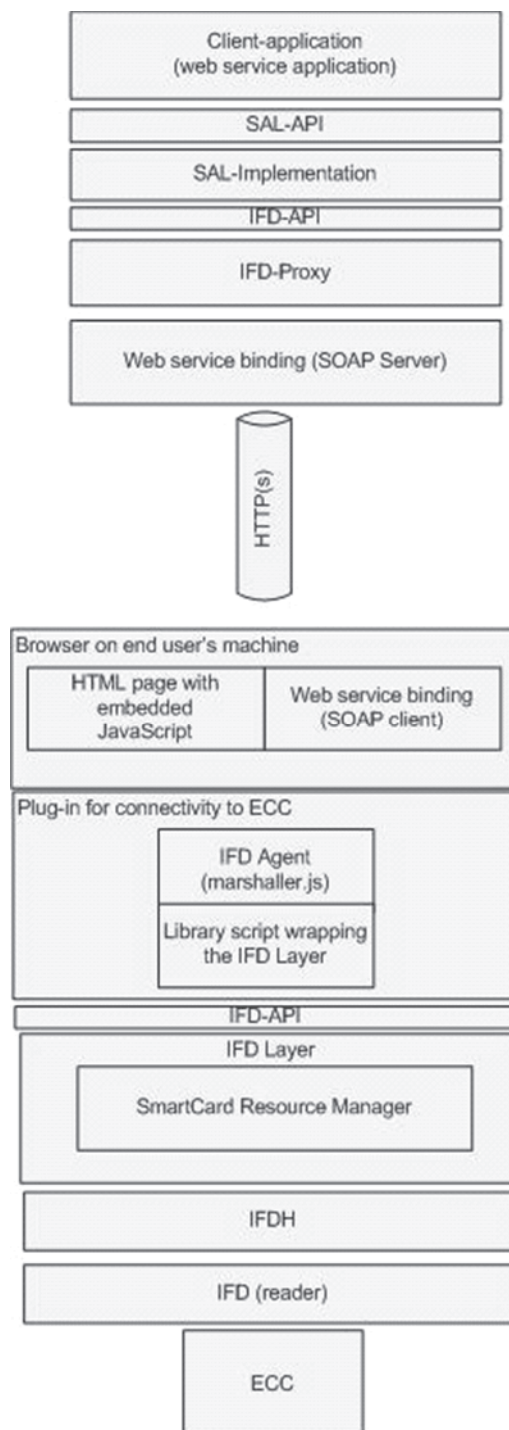


Figure 10 — Middleware architecture based on smartcard connectivity solution

The main advantages of this alternative are as follows:

- no installation required on end user's machine except the connectivity plug-in;
- light footprint scripts on end user's machine for the Browser extension;
- high level of mobility and interoperability: connection to the 24727-enabled web service from any PC;
- interoperability of transaction based on IFD-Web Service Binding (Clause 8).

5.3.6 XML-based SAL interface

The present document specifies a new function available to the Client-Application: '**ExecuteSAL**' defined as a single entry point to spare ASN.1 encoding/decoding to the client-application and to convey a XML-based payload instead. This XML-based payload is conform to the SAL-interface defined in ISO/IEC 24727-3 and may be used in Web Service based scenarios or with the `ExecuteSAL` function call.

The XML-Schema definitions for Service Access Layer functions are provided in Clause 10. The C-Language binding for the `ExecuteSAL` function is provided in Annex G. The Java-Language binding for the `ExecuteSAL` function is provided in Annex H.

To ensure conformance of Authentication Protocols to the ISO/IEC 24727 environment, a template for protocol metadata definition is provided in ISO/IEC 24727-3:2008, Annex A. In ISO/IEC 24727-3:2008, Annex A, the authentication protocols binding is ASN.1 based, therefore an additional XML-Binding for these authentication protocols is required in order to make them fit with the `ExecuteSAL` function. Accordingly, examples of XML-Binding for a set of ECC-compliant authentication protocols is provided herein in Annex I.

5.3.7 Smart card profile fitting with ECC-3 stack

The figure below describes how ECC-compliant cards with profiles according ECC-4 can fit in ECC-3 middleware stack. Legacy interfaces as PKCS#11 or CryptoAPI are considered respectively for PKCS#11 enabled applications or Crypto API enabled applications.

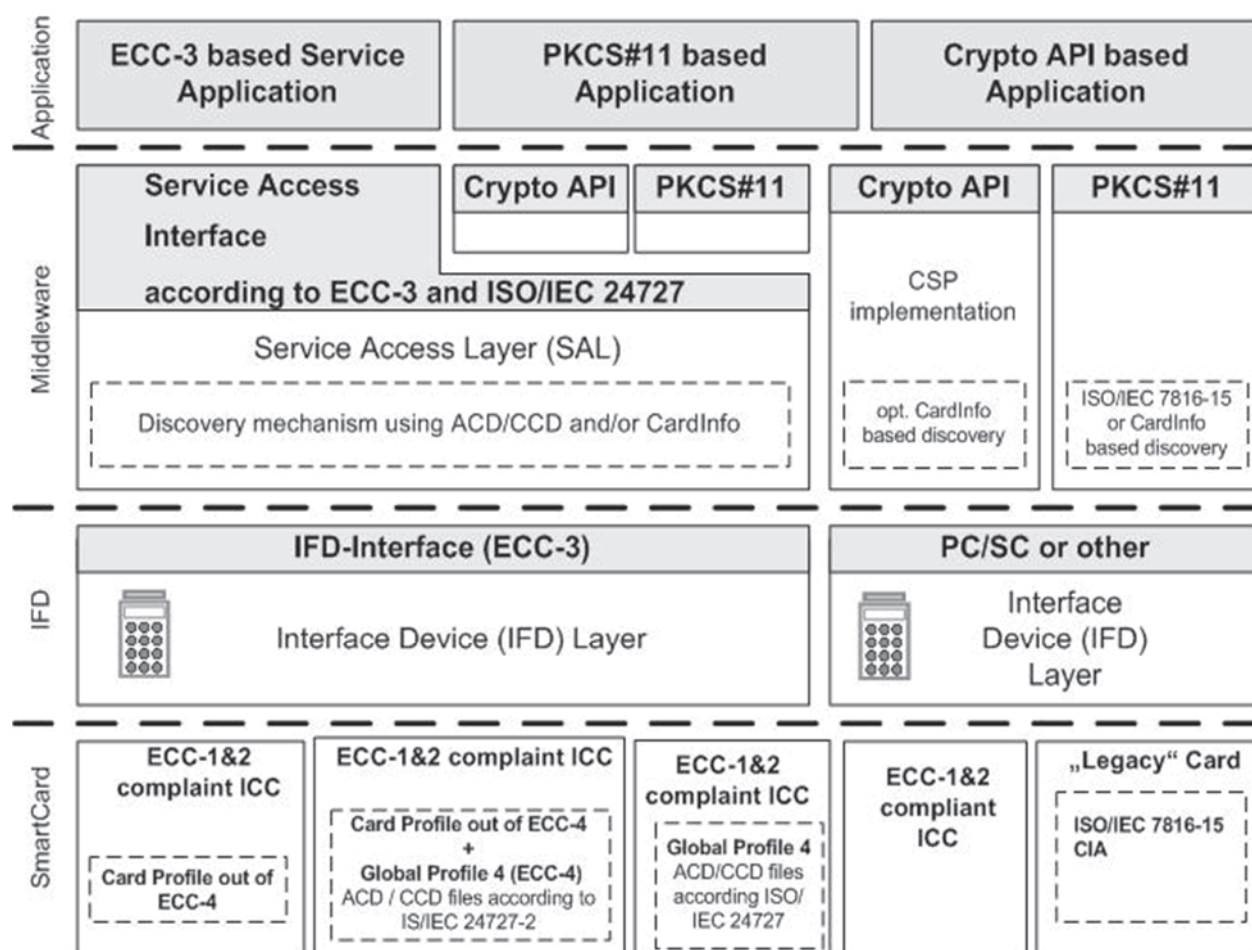


Figure 11 — ECC-compliant and legacy cards fitting with ECC-3 middleware stack

6 Card Discovery Mechanisms

The ECC-3 middleware supports multiple characteristics of ECCs. To adapt the SAL commands, the protocols and the data format to the card, the data structures described in 6.3 are used. If none of these data structures is present, the ECC middleware offers a mechanism to identify the card and to use configuration files for the command, data and protocol mapping. For identifying the card, different card specific data like AIDs, ACD, CCD and ISO/IEC 7816-15 files are used. A possible inspection activity chain is visible in Figure 12. To optimize the card recognition time, the order of the investigation can depend on the ATR or other optimization methods of the middleware.

6.1 Discovery decision tree

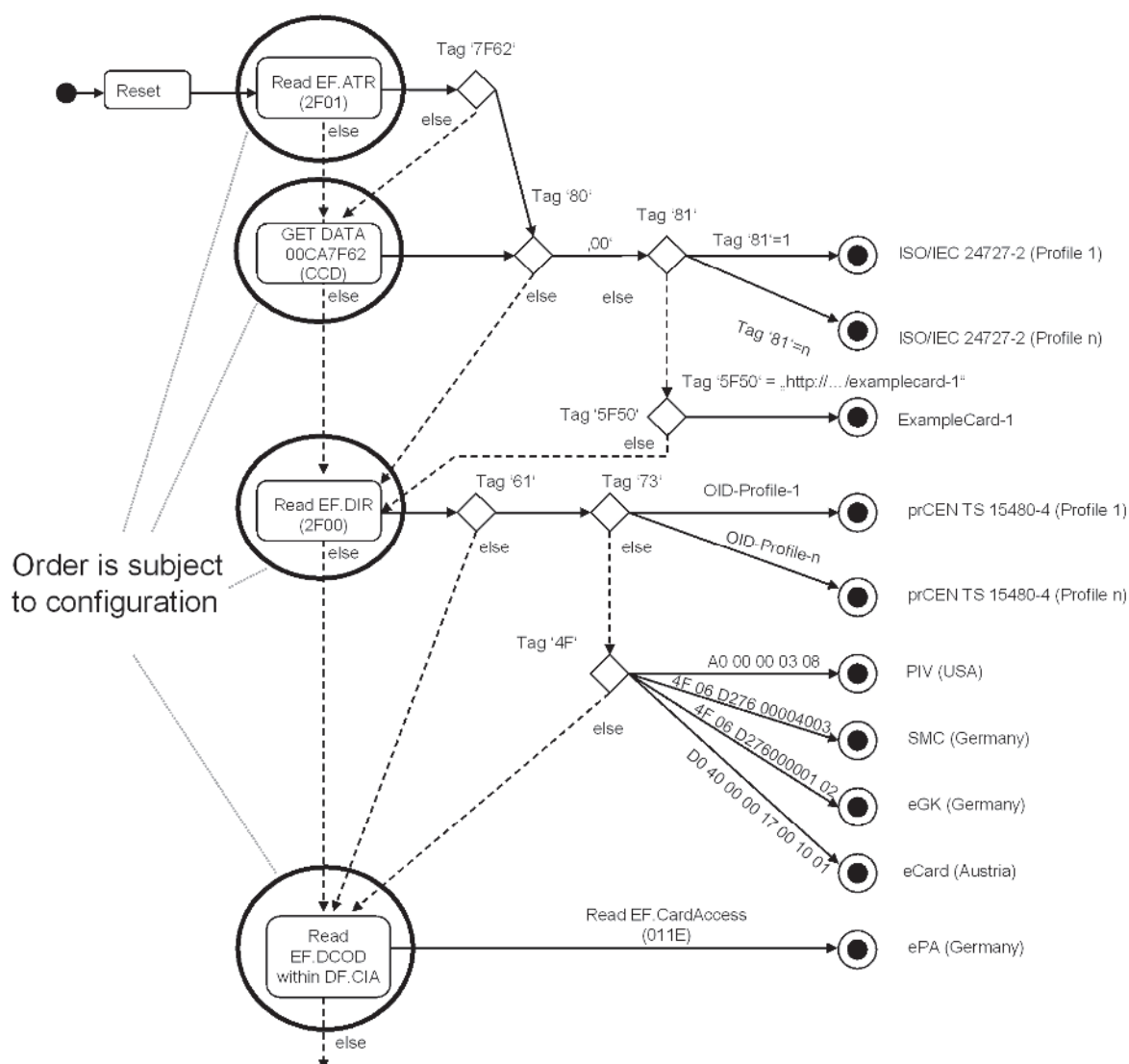


Figure 12 — Discovery decision tree

6.2 Migration path towards ECC and provision for legacy cards

This document describes the requirements for ECC token to fit into the ISO/IEC 24727 framework and provides recommendations as to how other cards may be supported. These requirements bear upon the encoding and personalization of some data that are to be made available to the terminal during the bootstrap procedure. According to ISO/IEC 24727-2, this procedure requires the presence of CCD and ACD containers in the card. As ISO/IEC 24727-non-compliant ID-cards and other proprietary cards are already deployed in the field, their personalization in post-issuance with additional data (i.e. CCD/ACD) might be very costly or impossible. The present document provides recommendations to allow for support of “legacy cards”.

For this purpose, it is necessary that the middleware:

- first takes provisions to recognize the type of a presented card and
- then obtains the CardInfo-file, which is required for the mapping of generic SAL-requests to card-specific APDUs by other means.

In order to implement the first step there are two alternatives:

- if the card contains a minimum CCD, the SAL will retrieve the full CardInfo-file at the address (URL) indicated in data object ‘5F50’ of the CCD;
- in order to support card which does not contain such a minimum CCD, the SAL shall be equipped with minimal XML-based CardInfo-files (cf. Clause 9), which just contain enough information to build the decision tree depicted in Figure 12. For each legacy card, which is to be supported, the CardType- and CardIdentification-element shall be described. Within the CardType-element there may be a CardInfoRepository-element, which provides the address (URL) of a repository.

After this procedure, the type of the card and/or the URL, which allows retrieving the CardInfo-file have been determined. There are the following possibilities to implement this second step:

- the SAL retrieves ASN.1-based ACD- and CCD-files from a remote server;
- the SAL uses the CardCapabilities- and ApplicationCapabilities-elements of the locally available XML-based CardInfo-file, which was used to build the decision tree in the first step;
- the SAL retrieves a full XML-based CardInfo-file from a remote server.

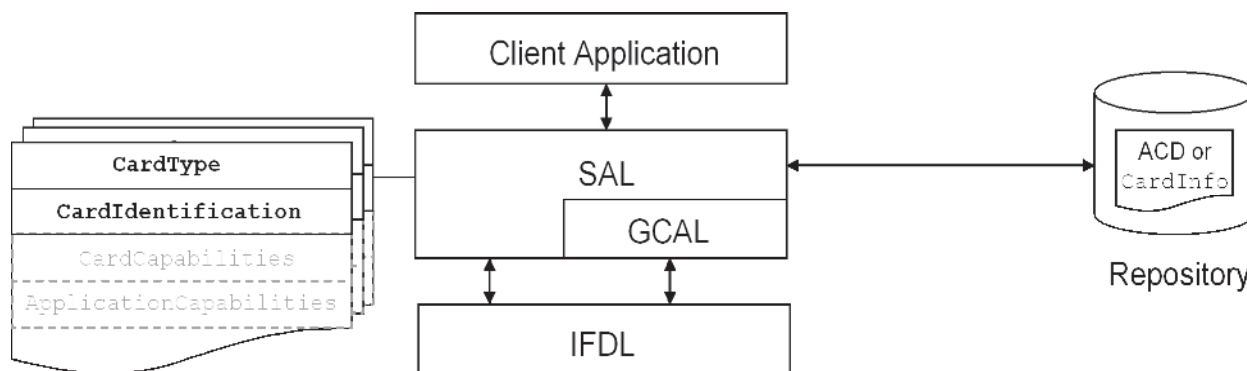


Figure 13 — Local or remote retrieval of ACD-like information

6.2.1 Interoperable access to the Repository

To ensure interoperable access from the SAL-implementation layer to the remote repository hosting ACD of CardInfo file containers, an XML-based transmission protocol is defined. The related Web Service Binding protocol is SOAP-based.

6.2.1.1 Definition of CardInfoRepository.XSD

The basic types and messages are defined in this section. The call to the remote repository is performed with a GetCardInfoOrACD function.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  targetNamespace="urn:iso:std:iso-iec:24727:tech:schema"
  xmlns:iso="urn:iso:std:iso-iec:24727:tech:schema"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
  elementFormDefault="qualified" attributeFormDefault="unqualified"
  >

  <include schemaLocation="CardInfo.xsd" />

  <!-- =====
        Interface to the CardInfoRepository
        ===== -->

  <!-- ===== -->
  <!-- Definition of CapabilityInfo -->
  <!-- ===== -->

  <element name="CapabilityInfo" type="iso:CapabilityInfoType" />
  <complexType name="CapabilityInfoType">
    <sequence>
      <element name="CardTypeIdentifier" type="anyURI" />
      <element name="CCD" type="hexBinary" maxOccurs="1"
        minOccurs="0" />

      <element name="ACD" type="hexBinary" maxOccurs="unbounded"
        minOccurs="1" />
    </sequence>
  </complexType>

  <!-- ===== -->
  <!-- GetCardInfoOrACD -->
  <!-- ===== -->

  <element name="GetCardInfoOrACD">
    <complexType>
      <sequence>
        <element name="CardTypeIdentifier" type="anyURI"
maxOccurs="unbounded" minOccurs="0"/>
        <element name="Action" type="anyURI" maxOccurs="1" minOccurs="0"/>
      </sequence>
    </complexType>
  </element>

  <element name="GetCardInfoOrACDResponse">
    <complexType>
      <complexContent>
```

```

        <extension base="iso:ResponseType">
            <choice maxOccurs="unbounded" minOccurs="0">
                <element name="CardInfo"
                    type="iso:CardInfoType"/>
                <element name="CapabilityInfo"
                    type="iso:CapabilityInfoType"/>
            </choice>
        </extension>
    </complexContent>
</complexType>
</element>

</schema>

```

6.2.1.2 Defintion of CardInfoRepository.WSDL

This section defines the SOAP-based Web Service Binding for CardInfoRepository-API.

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="urn:iso:std:iso-iec:24727:tech:schema"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:iso="urn:iso:std:iso-iec:24727:tech:schema"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    >

    <!-- ===== -->
    <!-- Definition of types -->
    <!-- (only include XSDs) -->
    <!-- ===== -->

    <wsdl:types>
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified"
            targetNamespace="urn:iso:std:iso-iec:24727:tech:schema">
            <xsd:include schemaLocation="CardInfoRepository.xsd" />
        </xsd:schema>
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        </xsd:schema>
    </wsdl:types>

    <!-- ===== -->
    <!-- Definition of messages -->
    <!-- ===== -->

    <wsdl:message name="GetCardInfoOrACDRequest">
        <wsdl:part name="parameters" element="iso:GetCardInfoOrACD" />
    </wsdl:message>
    <wsdl:message name="GetCardInfoOrACDResponse">
        <wsdl:part name="parameters" element="iso:GetCardInfoOrACDResponse" />
    </wsdl:message>

    <!-- ===== -->
    <!-- Definition of portType -->
    <!-- ===== -->

    <wsdl:portType name="GetCardInfoOrACD">
        <wsdl:operation name="GetCardInfoOrACD">

```

```
        <wsdl:input message="iso:GetCardInfoOrACDRequest" />
        <wsdl:output message="iso:GetCardInfoOrACDResponse" />
    </wsdl:operation>
</wsdl:portType>

<!-- ===== -->
<!-- Definition of Binding -->
<!-- ===== -->

<wsdl:binding name="GetCardInfoOrACD" type="iso:GetCardInfoOrACD">
    <soap:binding style="document"
        transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="GetCardInfoOrACD">
        <soap:operation
            soapAction="urn:iso:std:iso-iec:24727:tech:schema:GetCardInfoOrACD"
        />
        <wsdl:input>
            <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal" />
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>

<!-- Definition of CardInfoOrACD-Service -->

<wsdl:service name="GetCardInfoOrACD">
    <wsdl:port name="GetCardInfoOrACD" binding="iso:GetCardInfoOrACD">
        <soap:address location="http://127.0.0.1:18080" />
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

6.3 Set of data for interoperability

This clause defines the standard data structures to be personalized on the ECC card for compliance with ISO/IEC 24724. These data structures are intended to hide the low-level implementation features to the client-application. This section will refer to:

- application and Card Capabilities descriptors;
- ISO/IEC 7816-15 implementation including EF.DIR;
- any other information container located either on-card or off-card to achieve the fit with ISO/IEC 24727 framework (i.e. CardInfo-file, other).

6.4 Application and Card Capability Descriptors

Table 3 — Card Capability Descriptor options for ECC-compliant card describes the options required to encode ACD and CCD for ECC-compliant cards are assumed to host both CCD and as many ACD as interoperable application present on card. For each card-application which is not made available to a ISO/IEC 24727-compliant terminal, no ACD is required.

Table 2 — Application Capability Descriptor options for ECC-compliant card

Requirements for ECC-compliant cards				Comments
CCD with DO '7F62'				
Data Object	Tag	description	value	
PRO	'80'	24727-2 profile	'00'	Current version of the 24727-2 profile
SAID	'A0'	AID list	AIDs or empty	all card-applications made available to the eService via the SAL-API shall be referenced here
LANG	'A1'	Procedural elements	Empty	The mandatory set of APDU of ECC Part2 does not cover the whole set of commands of the Generic Card Interface (GCI) as per ISO/IEC 24727-2. All APDU commands likely to require procedural elements customization shall be locked
LANG-URL	'5F50'	URL to the full CardInfo-file	URL	the address of the CardInfo-file resource is present within a minimum CCD.
CIA-PROFILES	'81'	CIA profiles present on GCI	Appropriate Bit String	Optional: the card may implement a DF.CIA for the needs of the discovery mechanism (i.e. CIOChoice values are encoded as PathOrObject. If encoded as Path, CIOChoice items are nested in DF.CIA files, otherwise, they are encoded as Objects within the CardApplicationServiceDescription value. The DF.CIA may serve for further purposes (e.g. for any other application not interoperable in ISO/IEC 24727 environment).
CIA-PROFILES	'82'	CIA profiles present on GCI. Data structure for interoperability shall be made available to ISO/IEC 24727-2 layer		Optional: see above
DIGITAL-SIGNATURE-ON-CODE	'5F3D'	Digital signature information for procedural elements	Data object of digital signature block	Depending on security policy

Table 3 — Card Capability Descriptor options for ECC-compliant card

Requirements for ECC-compliant cards				Comments
ACD with DO '7F63' (may be empty)				
Data Object	Tag	description	value	
LANG	'A1'	Procedural elements	Empty	the present standard does not provide procedural elements implementation for ECC
LANG-URL	'5F50'	URL of the translation code	Empty	See above
SERVICE-DESCRIPTION	'7F66'	Services supported by the card-application	DER-TLV encoded CIAInfo and CIOChoice values according ISO/IEC 7816-15	discovery data are generated on the fly by the SAL out of the '7F66' data object content.
SERVICE-DESCRIPTION-LOCATION	'7F67'	Location of the '7F66' data objects	URL value	In case the '7F66' data object content is stored on a remote server, the '7F67' data object must provide the server resource location. In case the ACD or CCD are not present onto the card and are to be loaded from remote or provided by other means, they shall be complete ('7F67' data object does not apply)
DIGITAL-SIGNATURE-ON-CODE	'5F3D'	Digital signature information for procedural elements	Data object of digital signature block	Depending on security policy

6.5 ISO/IEC 7816-15 implementation

This clause describes the representation of ISO/IEC 24727-3 Card-Application discovery information as an ISO/IEC 7816-15 Cryptographic Information Application.

The ISO/IEC 7816-15 representation of an ISO/IEC 24727 card-application contains all the information that the ISO/IEC 24727-2, ISO/IEC 24727-3 implementations need to realize the ISO/IEC 24727-specified interoperable connection between the client-application and the card-application. Data structure for interoperability are discovery information made available to the SAL by the card through the bootstrap mechanism, unless it is provided by other means. The SAL shall undertake the processing of the discovery information in order to generate on the fly the data structures defined in ISO/IEC 24727-3 as Access Control List, Differential-Identity, Data-Set and Data Structures for Interoperability (DSI), and Card-Application Services and Actions. These data structures shall be surfaced to the eService upon request through the SAL-API.

To allow for coding of access control rules conforming to ISO/IEC 7816-15 implementation, a mapping of SAL-API actions onto ISO/IEC 7816-15 *accessMode* attributes is described in this clause. This mapping is laid on the extension of *accessMode* according the second amendment of ISO/IEC 7816-15.

The information in the ISO/IEC 7816-15 representation of an ISO/IEC 24727 card-application is what is referred to informally in ISO/IEC 24727-1 as the *discovery information*.

6.5.1 Profile designation within EF.DIR

Whenever a ECC implements a Profile as per prCEN/TS 15480-4, it shall conform to the recommendation of this clause.

EF.DIR shall host the Profile Object Identifier assigned by ECC-4 rules. Data object '73' shall encapsulate Profile OID within the application template (DO '61'). Accordingly bit6 of first byte within *card service data* in historical bytes shall be set to '1' to denote that BER-TLV DO is present in EF.DIR

The OID branch is:

OID = 1.3.162.15480.4.x.y with x.y denoting the Profile number.

For instance, DER-TLV coding with **x** = 0 and **y** = 2 (Global Profile n°2) valuates to 0x43 81 22 F8 78 04 00 02, therefore the Data Object denoting the Profile shall be 06 08 43 81 22 F8 78 04 00 02.

The figure below shows an example of how a Profile shall be encoded in a EF.DIR.

Tag	L	Description	Value		
61	var	Application template (for ADF CIA)			
		Tag	L	Description	
		4F	var	AID of CIA	0xE828BD080F A0000000xxxxxxxxxx
		50	03	Application Label "CIA"	0x43 49 41 («CIA»)
		Tag	L	Description	
		73	var	CIODDO Proprietary object	0x06 08 43 81 22 F8 78 04 00 02

Tag	L	Description	Value		
61	var	Application template (for Profile ADF called 'IAS')			
		Tag	L	Description	
		4F	var	AID of ECC-4 application	0xA0000000xxxxxxxxxx
		50	03	Application Label " IAS"	0x49 41 53 (« IAS »)
		51	02	FID	0x5F00

Figure 14 — Example of EF.DIR content

6.5.2 ISO/IEC 24727-3 data structures mapping

The data structures that may be surfaced to the client-application upon request are DataSet, DSI, ACL, Differential-Identities, list of Card-Application Services, list of Differential-Identities.

The SAL has to generate these data structures out of the information available in CardApplicationServiceDescription defined as an ASN.1 SEQUENCE of ISO/IEC 7816-15 CIAInfo value along with consecutive ISO/IEC 7816-15 CIOChoice(s) value(s).

Example of CardApplicationServiceDescription value implementation applying ISO/IEC 24727-2 rules is provided in Annex D.

The interpretation of CardApplicationServiceDescription value in terms of ACL, Differential-Identities, Services, DataSet or DSI, is based on the mapping described in the following sub-clauses.

6.5.2.1 DataSet

The table below shows the mapping of a DataSet onto a DataContainerObjectChoice and describes how the DataSet's ACL can be derived from it.

Table 4 — Mapping DataSet to ACL

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	ACL items	
DataContainerObject-Choice iso7816DO commonObjectAttributes	commonObjectAttributes.label	DataSet Name	X	The label is ASCII encoded and shall valuate to ' DATA-SET ' used as a prefix denoting the target type, concatenated with the DataSet actual name.
	accessControlRule.accessMode	Action		Shall conform to Table 10 and Table 11.
	accessControlRule.securityCondition	Ref to DID	X ²	authId referring to a CIO within the cryptographic information application (DF.CIA)
ClassAttributes	classAttributes.applicationName			Either applicationName or application OID of the card-application owning the DataSet. Shall not be NULL.
TypeAttributes	<unused>			NULL

² In case the Action is controlled by a logical combination of DIDs, several consecutive securityCondition values may provide different security conditions applying to the same access Mode denoting the SAL-API Action. The logical operation combining the DIDs is AND or OR depending on the ASN.1 encoding of the securityConditions.

Table 4 (continued)

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Descripti on	ACL items	
iso7816DO container for the DSI(s) within the DataSet				
CommonObjectAttributes	commonObjectAttributes.label	DSIName		The DSI Name shall be ASCII encoded.
	accessControlRule.accessMode	<unused>		DSI's accessRules are controlled by DataSet
	accessControlRule.securityCondition	<unused>		DSI's accessRules are controlled by DataSet accessRules
ClassAttributes		<unused>		
TypeAttributes	efidOrTagChoice	location of the actual resource (EF,DF, ADF, DO) hosting the DSI content.		File Id or extended Path (acc. ISO/IEC 7816-15:2004/Amd 2) to the DSI resource

The DIDs referenced by the access rules protecting a DataSet are either Authentication Information Objects or Private Objects or Secret Key. Secret Key and Private Key mapping to Differential-Identities is presented below.

6.5.2.2 CardApplication

The table below shows the mapping of a CardApplication onto a DataContainerObjectChoice and describes how the CardApplication's ACL can be derived from it.

Table 5 — Mapping CardApplication to DataContainerObjectChoice

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3	Comments
		Description ACL items	
DataContainerObjectChoice iso7816DO commonObjectAttributes	commonObjectAttributes.label	Card-Application Name	X The label is ASCII encoded and shall valuate to ' CARD-APPLICATION ' used as a prefix denoting the target type, concatenated with the CardApplication actual name.
	accessControlRule.accessMode	SAL-API Action	
	accessControlRule.securityCondition	Ref to DID	X ³ May be authId referring to a CIO within the cryptographic information application (DF.CIA)
ClassAttributes	classAttributes.applicationName		Either applicationName or application AID. Shall not be NULL.
TypeAttributes	<unused>		NULL

³ In case the Action is controlled by a logical combination of DIDs, several consecutive securityCondition values may provide different security conditions applying to the same access Mode denoting the SAL-API Action. The logical operation combining the DIDs is AND or OR depending on the ASN.1 encoding of the securityConditions.

Table 5 (continued)

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Descripti on	ACL items	
iso7816DO container for each target resource of the CardApplication (DataSet Name, DID Name)	CommonObjectAttributes	DataSet or DID Name		The target name shall be ASCII encoded with a prefix denoting the target type concatenated with the actual target name.
	accessControlRule.accessMode	<unused>		Refer to the named target
	accessControlRule.securityCondition	<unused>		Refer to the named target
	ClassAttributes	<unused>		
TypeAttributes	efidOrTagChoice	location of the actual resource (EF,DF, ADF, DO) hosting the named target.		File Id or extended Path (acc. ISO/IEC 7816-15:2004/AM2) to the named target resource

6.5.2.3 Service

The table below shows a possible alternative for the mapping of a Service onto a DataContainerObjectChoice. But since a Named Service is not a target, no ACL is attached to it according ISO/IEC 24727. Deriving ACL from Named Service is not recommended by the present standard.

Table 6 — Mapping Service to ACL

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	ACL items	
DataContainerObject-Choice iso7816DO for the Service commonObjectAttributes	commonObjectAttributes.label	Service Name	X	The Service Name shall be comprised of ASCII coded prefix ' SERVICE ' concatenated with the respective Service Name conforming to ISO/IEC 24727-3 naming rules
	accessControlRule.accessMode	SAL-API Action		two supported Actions for Authorization Service : ACL_LIST, ACL_MODIFY
	accessControlRule.securityCondition	Ref to DID	X ⁴	encoding shall conform to Table 10 and Table 11. authId. ACL_LIST may valuate to ALWAYS. authId is the cross-reference to an Authentication Object protecting the current CIO

⁴ This ACL is for Authorization Service that applies for all card-application services.

Table 6 (continued)

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	ACL items	
ClassAttributes	applicationName			applicationName or/and application OID. Shall not be NULL. It gives the location of the executable code implementing the named Service within the card application.
	applicationOID			
TypeAttributes	<unused>			NULL
iso7816DO container for the Actions within the Service				
CommonObjectAttributes	commonObjectAttributes.label	<optional> Action Name		The Action Name shall conform to ISO/IEC 24727-3 naming rules. If present, it clears ambiguity for SAL-API actions sharing the same AccessMode byte value.
	accessControlRule.accessMode	SAL-API Action		Shall conform to Table 10 and Table 11
	accessControlRule.securityCondition	Ref to DID	ACL ⁵	authId. this is the cross-reference to an Authentication Object protecting the current CIO

⁵ This ACL relates to a specific Action within the card-application named Service

Table 6 (continued)

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	ACL items	
ClassAttributes	applicationName			The applicationName or/and application OID shall not be NULL. It gives the location of the executable code implementing the named Service within the card application.
	applicationOID			
	CommonDataContainerObjectAttributes.iD	Ref to a CIO		
				The iD may be used to associate the current data container object with some other CIO (e.g. a private key information object, a public key information object, a certificate information object, a secret key information object)
TypeAttributes	efidOrTagChoice	location of the cryptographic information object executing the Action (e.g. a PrK for a SIGN action)		File Id or Path to the CIA directory file containing the CIO referenced by the ClassAttributes.iD above

6.5.2.4 Differential-Identity

In the context of ISO/IEC 24727-3, Differential-Identities are used to describe objects for authentication and cryptographic usage. As a consequence, ISO/IEC 7816-15 AuthenticationObjectChoice structures are used to describe differential identities with the focus on authentication.

KeyObjectChoice structures will be used to qualify differential identities which should mainly be used for the cryptographic methods (Encipher, Decipher, Sign,...).

6.5.2.4.1 Differential-Identities for authentication

The structure of a differential identity is heavily depending on the used authentication protocol. The marker and qualifier structures are only known after the authentication protocol has been identified. Therefore, the

first step in the discovery mechanism is to retrieve the correct value for the authentication protocol while parsing the service description of the Application Capability Descriptor.

The discovery mechanism for the authentication protocol value is shown in Figure 15. The mechanism is starting with parsing the AuthenticationObjectChoice. At the beginning, the TypeAttributes of the AuthenticationObjectChoice will be determined. If the TypeAttributes are PasswordAttributes or BiometricAttributes, the type of the authentication protocol will be either the PIN Compare or the Biometric Compare protocol.

In case the TypeAttributes are ExternalAuthObjectAttributes, it has to be investigated which choice was made for the ExternalAuthObjectAttributes:

- if CertBasedAuthenticationAttributes are used, the protocol will be the Asymmetric External Authenticate protocol;
- in the case AuthKeyAttributes are used, it's the Symmetric External Authenticate Protocol unless otherwise specified by the related objId: the corresponding AlgorithmInfo within the CIAInfo structure of the adequate application will be used to retrieve the objId. If the equivalent non DER-TLV encoded value of this objId is starting with "1 0 24727 3", the value will be directly used to determine the authentication protocol value.

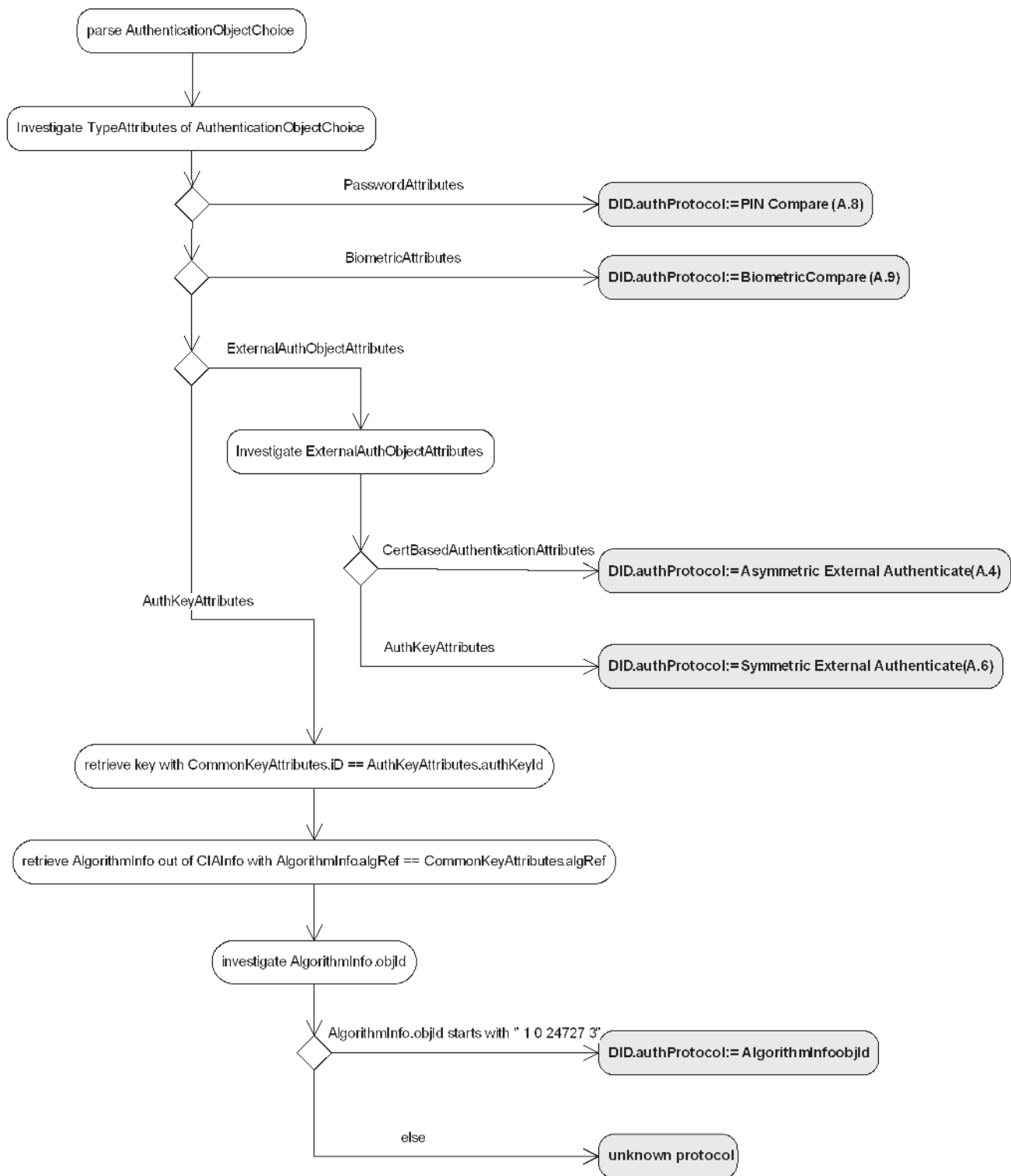


Figure 15 — Discovery of the authentication protocol value

The table below shows the mapping of an Authentication Object to the related Differential-Identity

Table 7 — Mapping authObject to DID

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	DID item	
AuthenticationObject-Choice commonObjectAttributes	commonObjectAttributes.label	Name denoting the authObject purpose (ASCII-encoded)	DIDName	
	commonObjectAttributes.authId	Cross-reference to the algorithm used to authenticate this DID.	Auth Protocol	Matches the value AlgorithmInfo.algRef in CIAInfo.supportedAlgorithms
	accessControlRule.accessMode	SAL-API Action		Shall conform to Table 10 and Table 11;
	accessControlRule.securityCondition	ISO/IEC 7816-15 compatible		Alternatively (not recommended), the access rules controlling the SAL-API Actions executable upon this DID may be found in Differential-Identity Service ACLs
ClassAttributes	authID	Cross-reference from any CIO to this DID. Shall match for instance authId used as securityConditions attribute for DataSets or Services.		<i>CommonAuthenticationObjectAttributes.authID</i> enables DID lookup for private objects or dataContainerObjects.

Table 7 (continued)

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	DID item	
	authReference	Key reference of the cryptographic object involved in the DID Auth Protocol in security environments	Marker	
	seldentifier	If present, it identifies the security environment to which the DID Marker and Auth Protocol belong.		
TypeAttributes	pwdFlags.local	If present (for pwd and biometric template CIOs), it denotes whether the object is local or global	Scope	
	Any other TypeAttribute shall comprise the Qualifier (i.g pwdType, minLength, storedLength, maxLength, padChar, path)	If present, shall be part of the Qualifier	Qualifier	once the OID of the authentication protocol is determined by the terminal, the actual Qualifier content shall conform to the definition provided in Annex A of ISO/IEC 24727 Part 3.

The Table below shows the mapping of a Secret Key to a Differential-Identity.

Table 8 — Mapping Secret Key to DID

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	DID Item	
SecretKeyChoice commonObjectAttributes	commonObjectAttributes.label	Name denoting the secretKey purpose (ASCII-encoded)	DIDName	
	commonObjectAttributes.authId	Cross-reference to the authentication object used to protect this secret key	Auth Protocol	May be employed as cross-reference to the algorithm used to authenticate this DID. If so, it shall matches the value AlgorithmInfo.alg Ref in CIAInfo.supported Algorithms. But it is recommended to use authId as a cross-reference to the authentication object used to protect this secret key
	accessControlRule.accessMode	SAL-API Action		Shall conform to Table 10 and Table 11.
	accessControlRule.securityCondition			Alternatively (not recommended), the access rules controlling the ISO/IEC 24727-3 Actions executable upon this DID may be found in Differential-Identity Service ACLs.

Table 8 (continued)

ISO/IEC 7816-15 item	ISO/IEC 7816-15 sub-attribute(s)	ISO/IEC 24727-3		Comments
		Description	DID Item	
ClassAttributes	iD	Unique identifier of the Secret Key in the ISO/IEC 7816-15 structure		Used as cross-reference from access rules of DataSets or Services. iD value shall match authId value.
	usage	Indicates the possible usage of the key		Denotes the usage authorized for the DID Marker
	native	<unused> Mandatory attribute but unused for DID building		Denotes whether the algorithm use by the key is implemented in the card hardware
	keyReference	Unique reference	Marker	Reference used in CRT (within security environments) or cryptographic commands to reference the key in the card-Application. Encoded as INTEGER
	algReference	identifies algorithms that may be used with the key by referencing supported-Algorithm values from CIAInfo.	Auth Protocol	Preferably used instead of CommonObject Attributes.authId . This attribute is Optional but shall be used for DID mapping.