Architecture of ID-software

Document version 0.5

Software version 3.11

Last updated: 01.06.2015

Table of contents

Introduction

The purpose of this document is to describe the architecture of ID-software.

ID-software is a collection of software components offering support for PKI-based functionality, i.e. operations with different cryptographic tokens (e.g. eID cards), handling digitally signed documents, file encryption/decryption and signing and authentication in web environment. The ID-software comprises end-user applications, software libraries, web components, drivers for communicating with the cryptographic tokens and other complementary components.

This document covers description of ID-software and its components, their deployment in different environments, provided and required interfaces. The document does not include components that have reached the end of their support nor the components that have not yet been released.

The document is based on the latest released state of the ID-software components. At the time of writing, the latest released version of ID-software is **version 3.11**. Latest version numbers of the various ID-software components are provided at http://www.id.ee/?lang=en&id=36798.

The document is targeted for:

1. Owners/managers of the software;
2. Contractors;
3. Contributors interested in developing ad-hoc solutions;
4. Integrators/software developers interested in integrating the software with third-party information systems;
5. International audience – contributors/integrators from countries other than Estonia who wish to use the software internationally and/or contribute in its development.

Background

The main owner/manager of the ID-software is Estonian Information System Authority (RIA, https://www.ria.ee/en/).

Main contractor for developing the software is AS Sertifitseerimiskeskus (SK, https://sk.ee/en). In case of a few of the components, SK is also the owner.

Development of ID-software has been mainly done in Estonia, however, the ID-software is released for international usage.

The software is distributed open-source (mainly under LGPL/BSD licence) and is accessible from the following locations:

1. GitHub repository for the source code and beta versions of binary packages: https://github.com/open-eid.
2. Release repository for binaries: https://installer.id.ee/

ID-software components can be logically divided in the following groups:

1. **Desktop applications** for end-users;
2. **Software libraries** for integrators/software developers to integrate the libraries' functionality with third-party information systems/applications;
3. **Web components** for integrators/software developers to add the signature creation and authentication functionality in web environment to third-party web applications;
4. **Drivers** for communication with the cryptographic tokens that conduct the PKI operations;
5. **Other (supportive) components** for packaging, installation, updating and version-control of the software.

The following table maps the ID-software components, their owner/developer (i.e. the main contractor) and the functionality they offer.

Table 1. Mapping of ID-software components and functions

	Component	Function					Owner/ Developer	Licence
		Handling DDOC/ BDOC documents	Handling CDOC documents	Calculating RSA signature	Card management operations	Authenti-cation		
Desktop applications	**DigiDoc3 Client**	+*	+	-	-	-	RIA/SK	LGPL
	ID-card utility	-	-	-	+*	-	RIA/SK	LGPL
Software libraries	**JDigiDoc (Java)**	+	+	+*	-	-	RIA/SK	LGPL
	DigiDoc4j (Java)	+	-	+*	-	-	RIA/SK	LGPL
	Libdigidocpp (C++)	+	-	+*	-	-	RIA/SK	LGPL
	CDigiDoc (C)	+	+	+*	-	-	RIA/SK	LGPL
	NDigiDoc (.NET)	-	+	-	-	-	SK/SK	BSD3
Web components	**Browser signing modules**	-	-	+*	-	-	RIA/SK	LGPL
	hwcrypto.js (JavaScript)	-	-	+*	-	-	RIA/SK	MIT
	pkcs11-module-loader	-	-	-	-	+**	RIA/SK	LGPL
Driver components	**Minidriver**	-	-	+*	-	+*	RIA/SK	LGPL/BSD3
	EstEID-pkcs11	-	-	+*	-	+*	RIA/SK	-
	EstEID-tokend	-	-	+*	-	+*	RIA/SK	APSL / LGPL
	Smartcardpp	-	-	+*	+*	+*	RIA/SK	LGPL/BSD3

* - The functionality is provided via base components

** - The component is used only once for setting the proper parameters for authentication in Firefox

browser.

The main functions offered by ID-software are described in the following table.

Table 2. Functions offered by ID-software

Function	Description
Handling DDOC/BDOC documents	Handling documents in BDOC 2.1 (XAdES/ASiC-E) and DIGIDOC-XML 1.3 (DDOC) digital signature formats that are profiles of ETSI XAdES (XML Advanced Electronic Signature) format. More information on the formats' life cycle can be found from http://www.id.ee/?lang=en&id=34336.
Calculating RSA signature	Calculating the RSA signature value in browser or desktop/server environment. The operation involves connecting with the signature token's driver, sending the data to be signed and receiving digital signature value calculated with the token owner's RSA private key. The following cryptographic tokens are supported: 1. Hardware-based tokens (e.g. PKCS#11-based eID cards, USB cryptostick and Mobile-ID) 2. Software-based tokens (e.g. PKCS#12 software token)
Handling CDOC documents	Encrypting and decrypting documents in ENCDOC-XML 1.0 (CDOC) format.
Card management operations	Renewal of the certificates on the card, PIN/PUK management, reading personal data file.
Authentication	Authentication with ID-card. The operation is generally done via native operating system/browser components. In case of Estonian ID-cards and Firefox browser, a PKCS#11 module loader script is used for setting the proper parameters for authentication in Firefox browser.

Component model

The following chapter depicts ID-software component diagrams, including variations of the components used in different supported environments.

In the context of the component diagrams in this document, the ID-software components have been divided to three different packages to show the component's owner/developer:

1. Components of ID-software that are owned by RIA and developed by SK: placed in "RIA/SK" package;
2. Components of ID-software that are owned and operated by RIA: placed in "RIA" package;
3. Components of ID-software that are owned and developed by SK: placed in "SK" package.

Other components are regarded as external to ID-software.

Note that not all of the external base libraries are included in the component model to avoid duplicity with other documentation – the base libraries are listed and described in the documentation of the respective ID-software components and can be accessed via the references provided.

Desktop applications

DigiDoc3 Client

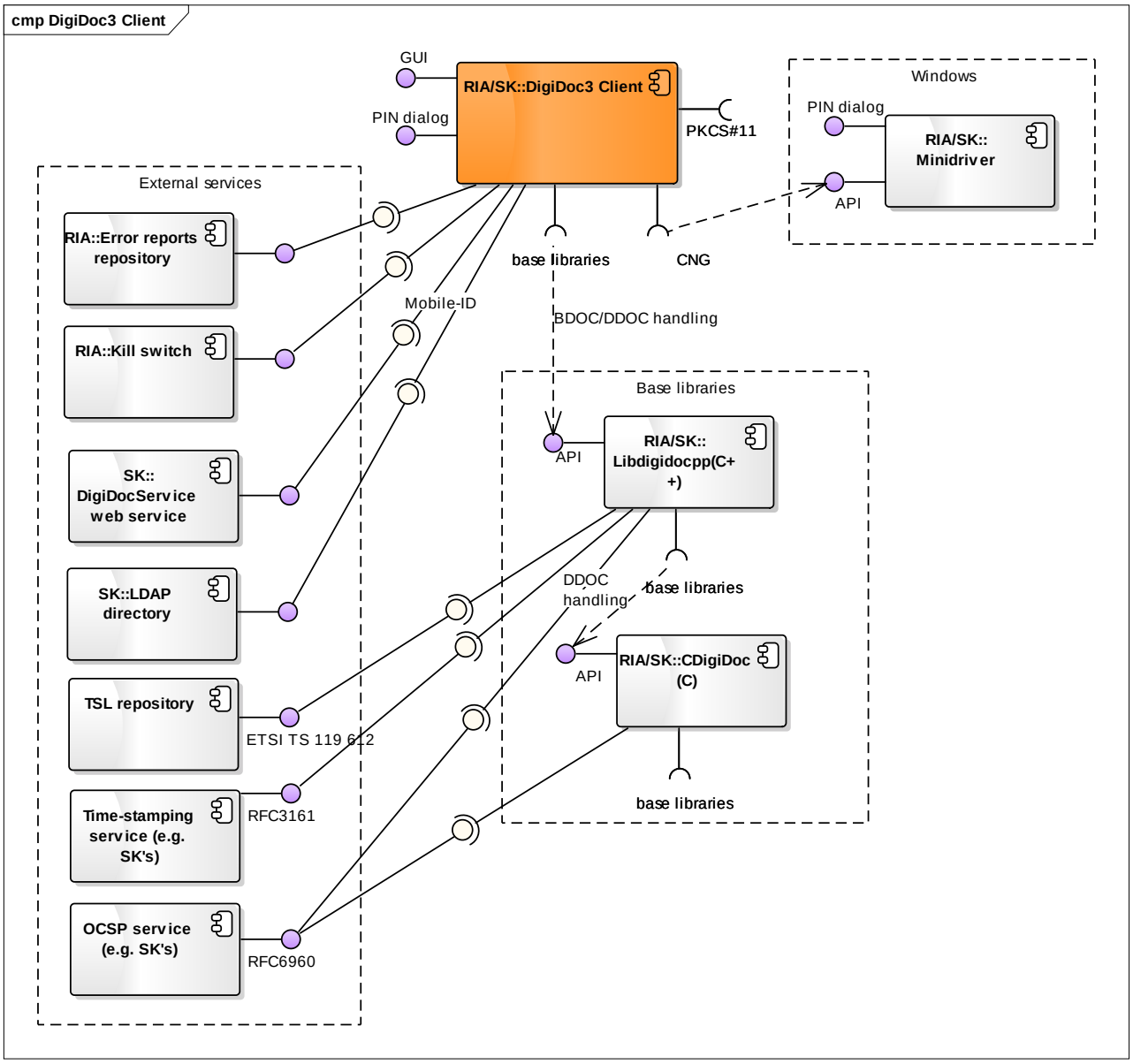

Figure 1. Components of DigiDoc3 Client

Table 3. Components of DigiDoc3 Client

Component	Description	Owner/ Developer
DigiDoc3 Client	End-user desktop applications that own a common GUI. DigiDoc3 Client enables handling digitally signed documents. DigiDoc3 Crypto subcomponent enables file encryption/decryption. Wiki: https://github.com/open-eid/qdigidoc/wiki Code repository: https://github.com/open-eid/qdigidoc	RIA/SK
DigiDoc3 Client base libraries	Libdigidocpp (and its base libraries, including CDigiDoc), etc. See ID-card utility interfaces	-

Component	Description	Owner/ Developer
Error reports repository	Repository where the DigiDoc3 Client application's and ID-card utility program's error reports (generated with BreakPad base library) are sent.	RIA
Kill switch	Service for centrally managing DigiDoc3 Client application's life cycle. The application periodically connects with the service to check if the application's version is still supported. If not, then the application cannot be used any longer and a newer version must be installed.	RIA
DigiDocService web service	SOAP-based web service that is used by DigiDoc3 Client for signature creation with Mobile-ID. See also http://www.sk.ee/upload/files/DigiDocService_spec_eng.pdf.	SK/SK
LDAP directory	Directory of active certificates issued by SK (as the CA in Estonia). The directory is used by DigiDoc3 Crypto subcomponent for finding authentication certificate (and the respective public key) of the recipient of the encrypted document. See also https://sk.ee/en/repository/ldap/ldap-kataloogi-kasutamine/	SK/SK
TSL repository	Repository for accessing the TSL (Trust Service status List) lists that can be used as a central source of trust anchor information during digital signature creation and validation processes. The European Commission's TSL list (https://ec.europa.eu/information_society/policy/esignature/trusted-list/tl-mp.xml) is used as the central TSL list (with references to national lists).	-
OCSP service	RFC6960 based OCSP service. Also offered by SK for Estonian and a number of foreign certificates (see www.sk.ee).	-
Libdigidocpp	Described in chap. Software libraries	RIA/SK
CDigiDoc	Described in chap. Software libraries	RIA/SK
Minidriver	Used via CNG interface in Windows environment only. Described in chap. Drivers	RIA/SK

DigiDoc3 Client interfaces

Provided:

1. Graphical user interface - interface for handling ASiC-E/XAdES (i.e. BDOC), DDOC, CDOC documents, setting configuration parameters.
    a. User: end-user
    b. Accessible with: GUI elements
2. PIN dialog – for inserting PIN value during signature creation or decryption operations in all operating systems except of Windows
    a. User: end-user
    b. Accessible with: GUI elements

Required:

1. Kill switch service interface
2. DigiDocService web service interface
3. Error reports repository interface

4. LDAP directory interface
5. Interfaces with base libraries:
    a. Libdigidocpp library's API – for handling documents in supported digital signature formats (BDOC and DDOC).
    b. External base libraries: Qt5, libldap, openssl
6. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. PKCS#11 interface
    b. CNG interface

ID-card utility

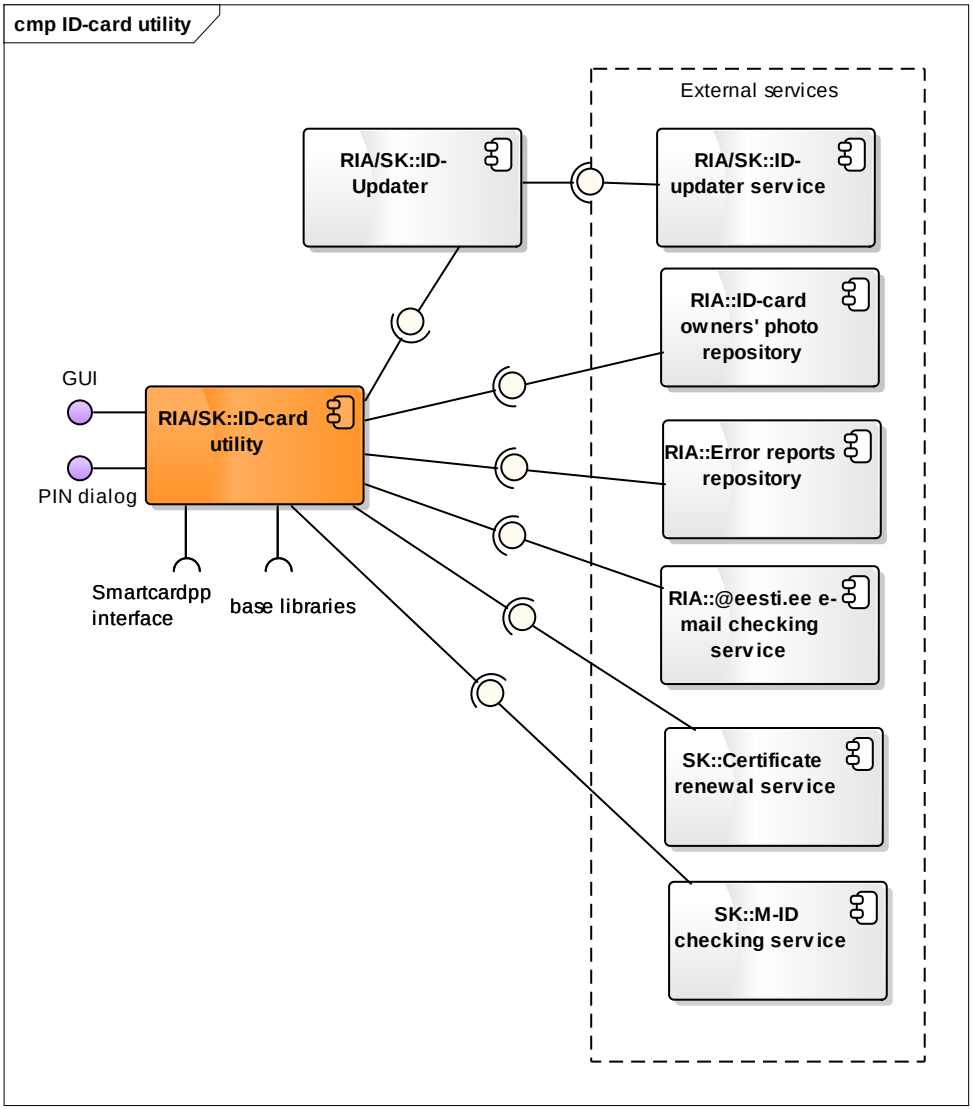

Figure 2. Components of ID-card utility

Table 4. Components of ID-card utility

Component	Description	Owner/ Developer
ID-card utility	End-user desktop application for managing ID-card's PIN/PUK codes replacement, certificates' renewal and other services. Code repository: https://github.com/open-eid/qesteidutil Wiki: https://github.com/open-eid/qesteidutil/wiki	RIA/SK
ID-card utility's base libraries	See ID-card utility interfaces	-
ID-card owner's photo repository	Repository where the Estonian national ID-cards photos' are kept. ID-card's owner can download the photo after the user has been authenticated with PIN1 code.	RIA
Error reports repository	Described in chap. DigiDoc3 Client	RIA

Component	Description	Owner/ Developer
@eesti.ee e-mail checking service	Service that enables to set the properties of e-mail address (@eesti.ee) that is provided for Estonian national ID-card owners by the state. The user must be authenticated with PIN1 code.	RIA
Certificate renewal service	Service for renewing certificates on the Estonian national ID-card.	SK
M-ID checking service	Service for checking the status of Estonian national ID-card owner's Mobile-ID certificates. The user must be authenticated with PIN1 code.	SK
Updater, ID-updater Service	Described in chap. Updating mechanisms	RIA/SK

ID-card utility interfaces

Provided:

1. Graphical user interface – interface for handling card management operations and using the external services (listed under "Required interfaces").
    a. User: end-user
    b. Accessible with: GUI elements
2. PIN dialog – for inserting PIN/PUK value in all supported operating systems.
    a. User: end-user
    b. Accessible with: GUI elements

Required:

1. ID-card owners' photo repository interface
2. Error reports repository interface
3. Certificate renewal service interface
4. Eesti.ee e-mail checking service interface
5. Certificate renewal service interface
6. Mobile-ID validity checking service interface
7. Interfaces with base libraries: Qt5
8. Interfaces with cryptographic token's drivers:
    a. Smartcardpp API (internal component)

Software libraries

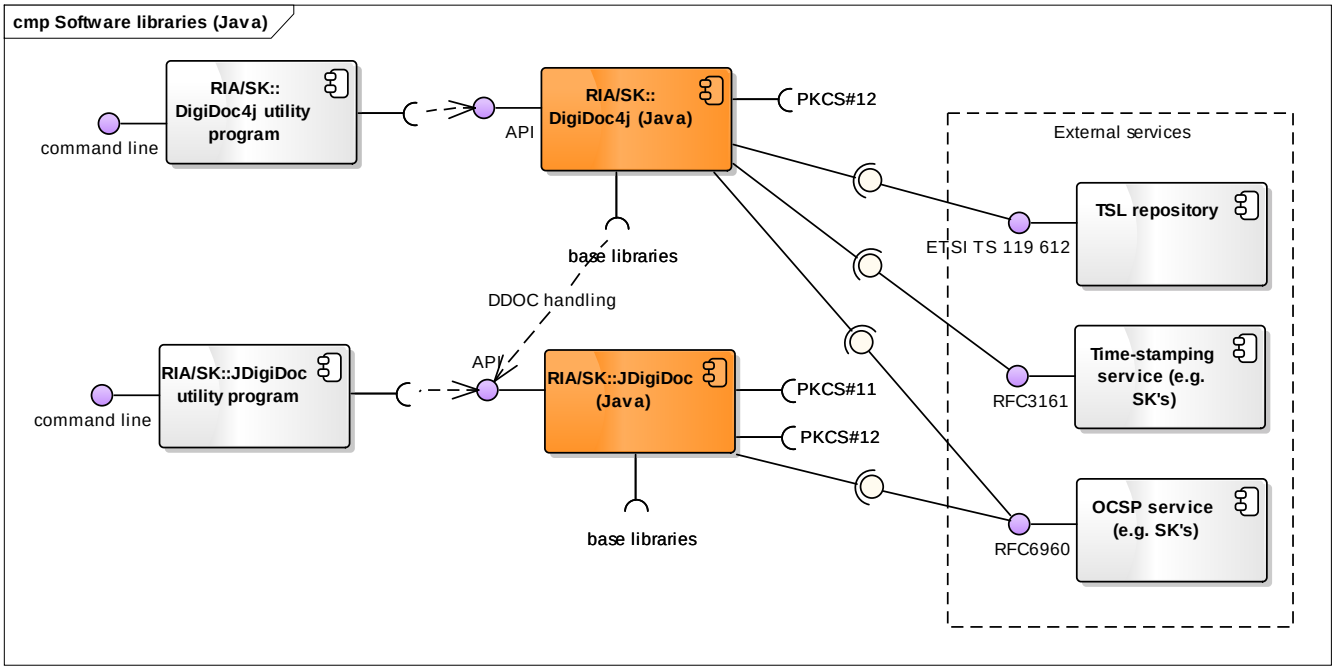

Figure 3. Java software libraries and their components

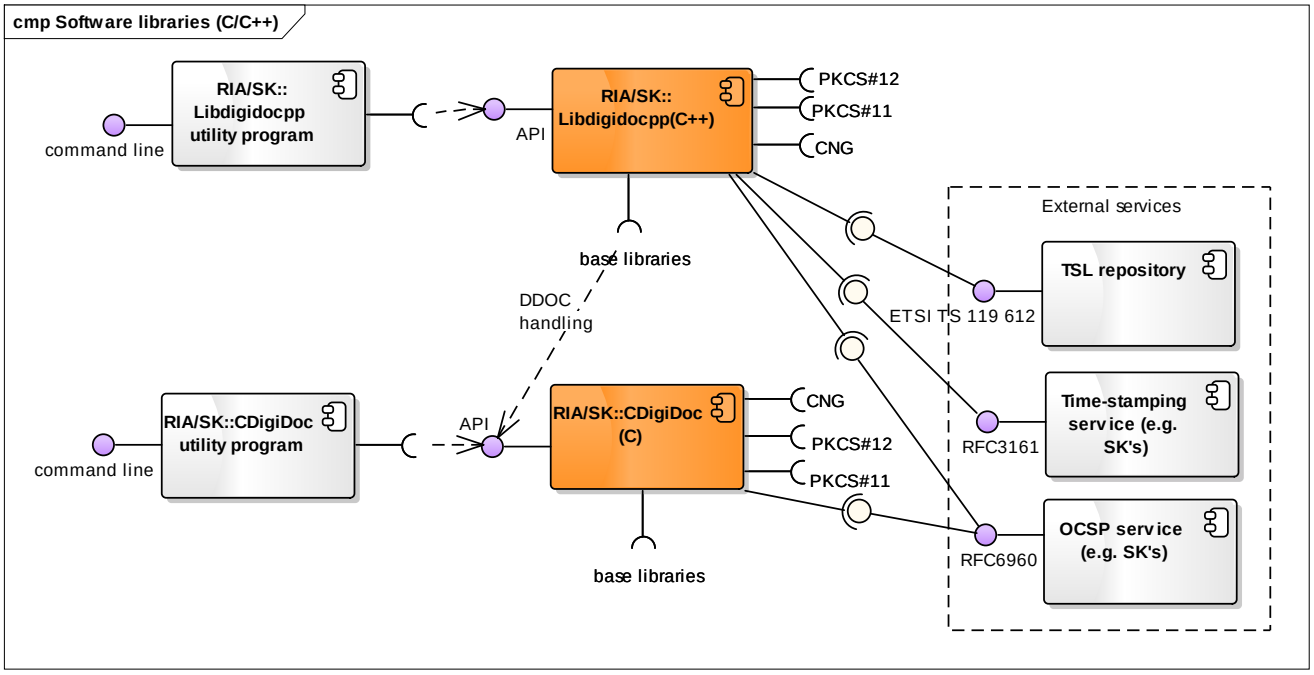

Figure 4. C/C++ software libraries and their components

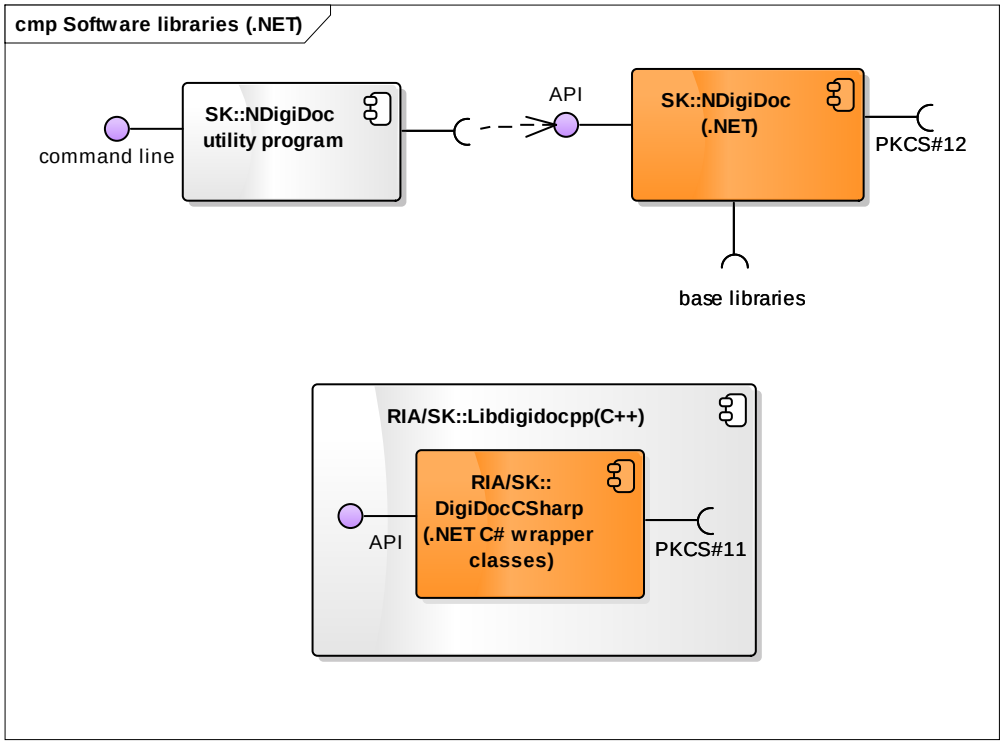

Figure 5. .NET software libraries and components

Table 5. Software libraries and their components

Component	Description	Owner/ Developer
DigiDoc4j	Java software library that enables handling documents in BDOC 2.1(XAdES/ASiC-E) and DIGIDOC-XML 1.3 formats. Documentation: http://open-eid.github.io/digidoc4j/ Code repository: https://github.com/open-eid/digidoc4j	RIA/SK
DigiDoc4j utility program	Small command line application that implements the main functionality of DigiDoc4j library. Used for testing purposes. Can also be used as a source for sample client code for using DigiDoc4j. See also http://open-eid.github.io/digidoc4j/	RIA/SK
JDigiDoc	Java software library that enables handling documents in BDOC 2.1 (XAdES/ASiC-E) and DIGIDOC-XML 1.3 formats and encryption/decryption in ENCDOC-XML 1.0 (CDOC). Documentation: http://id.ee/public/SK-JDD-PRG-GUIDE.pdf Code repository: https://github.com/open-eid/jdigidoc	RIA/SK
JDigiDoc utility program	Small command line application that implements the main functionality of JDigiDoc library. Used for testing purposes. Can also be used as a source for sample client code for using JDigiDoc. See also http://id.ee/public/SK-JDD-PRG-GUIDE.pdf.	RIA/SK
Libdigidocpp	C++ software library that enables handling documents in BDOC 2.1 (XAdES/ASiC-E) and DIGIDOC-XML 1.3 formats (via CDigiDoc base library). Wiki: https://github.com/open-eid/libdigidocpp/wiki Code repository: https://github.com/open-eid/libdigidocpp Documentation: http://open-eid.github.io/libdigidocpp/	RIA/SK

Component	Description	Owner/ Developer
Libdigidocpp utility program	Small command line application (digidoc-tool.exe) that implements the main functionality of Libdigidocpp library. Used for testing purposes. Can also be used as a source for sample client code for using Libdigidocpp. See also http://open-eid.github.io/libdigidocpp/	RIA/SK
CDigiDoc	Software library in C that enables handling digitally signed documents in DIGIDOC-XML 1.3 format and encryption/decryption in ENCDOC-XML 1.0 (CDOC). Documentation: http://id.ee/public/SK-CDD-PRG-GUIDE.pdf Code repository: https://github.com/open-eid/libdigidoc Wiki: https://github.com/open-eid/libdigidoc/wiki	RIA/SK
CDigiDoc utility program	Small command line application that implements the main functionality of CDigiDoc library. Used for testing purposes. Can also be used as a source for sample client code for using CDigiDoc. See also http://id.ee/public/SK-CDD-PRG-GUIDE.pdf	RIA/SK
NDigiDoc	Software library in .NET enabling encryption/decryption in ENCDOC-XML 1.0 (CDOC). Documentation: http://id.ee/public/NDigiDoc.pdf Code repository: https://github.com/open-eid/ndigidoc	SK/SK
NDigiDoc utility program	Small command line application that implements the main functionality of NDigiDoc library. Used for testing purposes. Can also be used as a source for sample client code for using NDigiDoc. See also http://id.ee/public/NDigiDoc.pdf.	SK/SK
DigiDocCSharp	.NET C# wrapper classes for using Libidigidocpp library's functionality in .NET environment. Created with Swig tool. See also https://github.com/open-eid/libdigidocpp/blob/master/examples/DigiDocCSharp/README.md	RIA/SK
TSL repository	Described in chap. DigiDoc3 Client	-
Time-stamping service	Described in chap. DigiDoc3 Client	-
OCSP service	Described in chap. DigiDoc3 Client	-

DigiDoc4j library's interfaces

Provided:

1. DigiDoc4j API
    a. User: DigiDoc4j utility program
    b. Accessible with: Java

Required:

1. TSL repositories' interfaces
2. Time-stamping service interface
3. OCSP service interface
4. Interfaces with base libraries:
    a. JDigiDoc library's API – for handling documents in DDOC format.
    b. Other base libraries: see http://open-eid.github.io/digidoc4j/.
5. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. PKCS#12 interface

DigiDoc4j utility program's interfaces

Provided:

1. DigiDoc4j utility program's interface
    a. User: server application, end-user application, end-user
    b. Accessible with: command line

Required:

1. DigiDoc4j API

JDigiDoc library's interfaces

Provided:

1. JDigiDoc API
    a. User: JDigiDoc utility program, DigiDoc4j library
    b. Accessible with: Java

Required:

1. OCSP service interface
2. Interfaces with base libraries: see http://id.ee/public/SK-JDD-PRG-GUIDE.pdf for more information.
3. Interfaces with cryptographic token's drivers (described in chap. Drivers):
    a. PKCS#11 interface
    b. CNG interface
    c. PKCS#12 interface

JDigiDoc utility program's interfaces

Provided:

1. JDigiDoc utility program's interface
    a. User: server application, end-user application, end-user
    b. Accessible with: command line

Required:

1. JDigiDoc API: see chap. JDigiDoc library's interfaces

Libdigidocpp library's interfaces

Provided:

1. Libdigidocpp API
    a. User: DigiDoc3 Client, Libdigidocpp utility program, DigiDocCSharp .NET wrapper classes
    b. Accessible with: C++

Required:

1. TSL repositories' interfaces
2. Time-stamping service interface
3. OCSP service interface
4. Interfaces with base libraries:
    a. CDigiDoc library's API – for handling documents in DDOC format. See chap. CDigiDoc library's interfaces
    b. Other base libraries: OpenSSL, xerces-c, xalan-c, codesynthesis-xsd, libxml-security-c. See also http://open-eid.github.io/libdigidocpp/
5. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. PKCS#11 interface
    b. CNG interface
    c. PKCS#12 interface

Libdigidocpp utility program's interfaces

Provided:

1. Libdigidocpp utility program's interface
    a. User: server application, end-user application, end-user
    b. Accessible with: command line

Required:

1. Libdigidocpp API: see chap. Libdigidocpp library's interfaces

CDigiDoc library's interfaces

Provided:

1. CDigiDoc API
    a. User: Libdigidocpp library, CDigiDoc utility program
    b. Accessible with: C

Required:

1. OCSP service interface
2. Interfaces with base libraries: OpenSSL, libxml2. See also http://id.ee/index.php?id=35782.
3. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. PKCS#11 interface
    b. CNG interface
    c. PKCS#12 interface

CDigiDoc utility program's interfaces

Provided:

1. CDigiDoc utility program's interface
    a. User: server application, end-user application, end-user
    b. Accessible with: command line/console

Required:

1. CDigiDoc API: see chap. CDigiDoc library's interfaces

NDigiDoc library's interfaces

Provided:

1. NDigiDoc API
    a. User: server application, end-user application; NDigiDoc utility program
    b. Accessible with: .NET

Required:

1. Interfaces with base libraries: see http://id.ee/public/NDigiDoc.pdf for more information.
2. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. PKCS#12 interface

NDigiDoc utility program's interfaces

Provided:

1. NDigiDoc utility program's interface
    a. User: server application, end-user application, end-user
    b. Accessible with: command line/console

Required:

1. NDigiDoc API: see chap. NDigiDoc library's interfaces

Web components

Web signing components

The web signing component diagrams describe components that are needed for signature creation in web applications with eID cards.

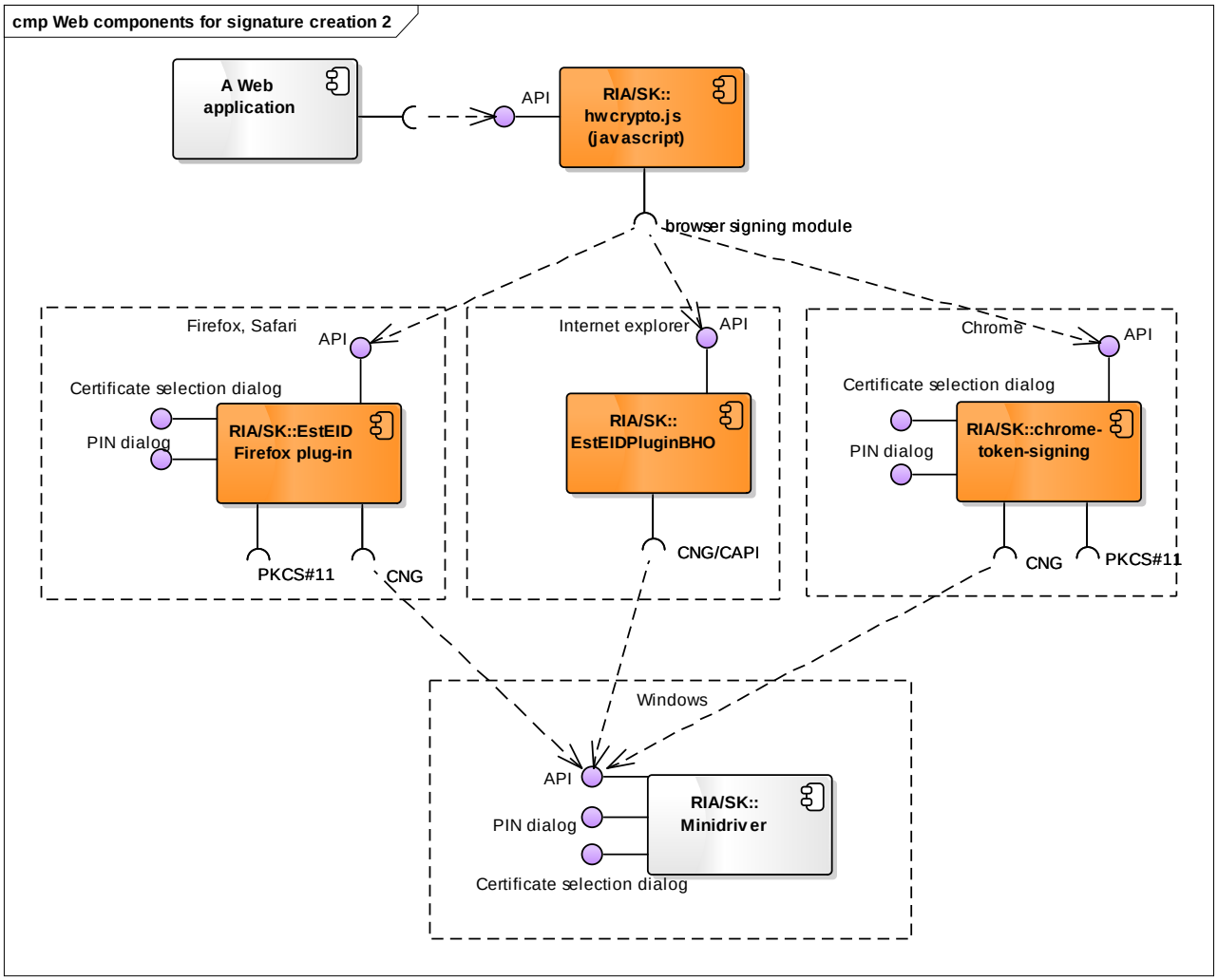

Figure 6. Components for signature creation in web environment

Table 6. Components for signing in web environment

Component	Description	Owner/ Developer
hwcrypto.js	JavaScript library that enables communication with the browser signing modules (plug-in or extension) of the different web browsers. Wiki: https://github.com/open-eid/hwcrypto.js/wiki Code repository: https://github.com/open-eid/hwcrypto.js	RIA/SK
A web application	A web application that implements signature creation with an eID-card in browser environment.	-

Component	Description	Owner/ Developer
EstEID Firefox plug-in	Browser signing module (NPAPI-based plug-in) that is used in Firefox (supported in Windows and Linux) and Safari (supported in Mac OS) browsers. The plug-in enables data exchange with the cryptographic token's driver that is used for signing. In Windows environment, the driver that is implementing CNG/CAPI interface is used, along with the operating system's native PIN insertion and certificate selection dialogs. Otherwise, PKCS#11 driver is used. Code repository: https://github.com/open-eid/browser-token-signing Wiki: https://github.com/open-eid/browser-token-signing/wiki	RIA/SK
EstEIDPluginBHO	Browser signing module (BHO-based plug-in) that is used in Internet explorer browser (supported in Windows operating system). The plug-in enables data exchange with the cryptographic token's driver that is used for signing. Code repository: https://github.com/open-eid/browser-token-signing Wiki: https://github.com/open-eid/browser-token-signing/wiki	RIA/SK
chrome-token-signing	Comprises two subcomponents: browser extension component and native OSX/Linux/Windows component that implements Native Messaging API (JSON). The browser extension enables data exchange with the native component that in turn interacts with the cryptographic token's driver for signing. Code repository: https://github.com/open-eid/chrome-token-signing Wiki: https://github.com/open-eid/chrome-token-signing/wiki	RIA/SK
Minidriver	Used via CNG interface in Windows environment only. Described in chap. Drivers	RIA/SK

Hwcrypto.js library's interfaces

Provided:

1. hwcrypto.js library's API
    a. User: a web application in browser environment
    b. Accessible with: JavaScript

Required:

1. Interfaces with browser signing modules:
    a. EstEID Firefox plug-in's interfaces
    b. EstEIDPluginBHO plug-in's interfaces
    c. Chrome-token-signing extension's interfaces

EstEID Firefox plug-in's interfaces

Provided:

1. EstEID Firefox plug-in's API
    a. User: a web application in browser environment, hwcrypto.js library
    b. Accessible with: C
2. PIN dialog – for inserting PIN2 value during signature creation in all operating systems except of Windows
    a. User: end-user
    b. Accessible with: GUI elements

3. Certificate selection dialog
    a. User: end-user
    b. Accessible with: GUI elements

Required:

1. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. PKCS#11 interface
    b. CNG interface

EstEIDPluginBHO plug-in's interfaces

Provided:

1. EstEIDPluginBHO plug-in's API
    a. User: a web application in browser environment, hwcrypto.js library
    b. Accessible with: C++

Required:

1. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. CNG/CAPI interface

Chrome-token-signing extension's interfaces

Provided:

1. Chrome-token-signing extension's API
    a. User: a web application in browser environment, hwcrypto.js library
    b. Accessible with: C++
2. PIN dialog – for inserting PIN2 value during signature creation
    a. User: end-user
    b. Accessible with: GUI elements
3. Certificate selection dialog
    a. User: end-user
    b. Accessible with: GUI elements

Required:

1. Interfaces with cryptographic token's drivers (described in chap. Drivers)
    a. PKCS#11 interface

Web authentication components

Authentication in web browsers is done with the browsers' and operating systems' native components. In case of authenticating in Firefox browser then Firefox pkcs11-module-loader JavaScript component is used to load the OpenSC PKCS#11 driver by the browser.

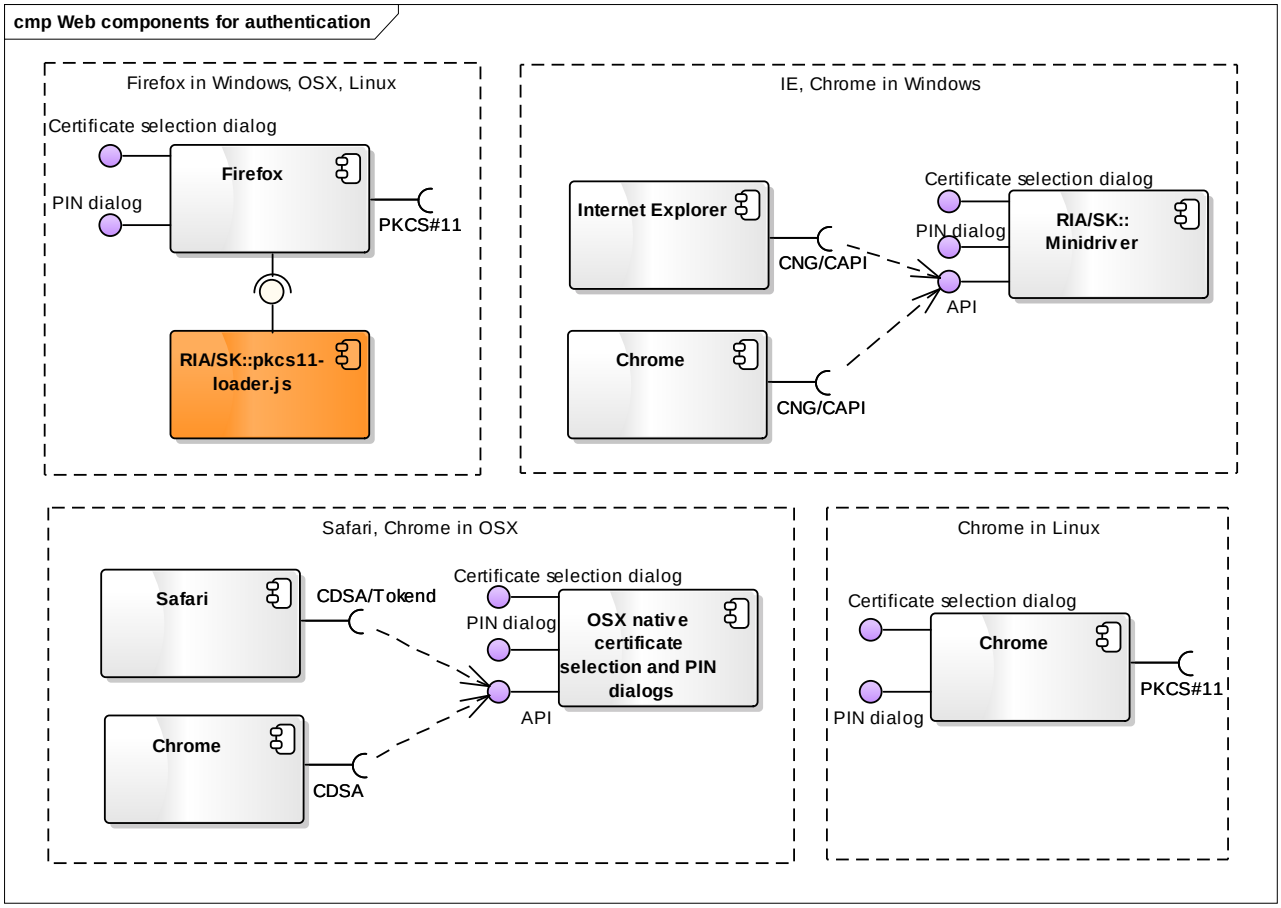

Figure 7. Web authentication components

Table 7. Web authentication components

Component	Description	Owner/ Developer
pkcs11-loader.js	A JavaScript component that is used to load the OpenSC PKCS#11 driver to the Firefox browser's cryptographic devices list during each initialization of the browser. Needed during authentication process with eID-card in Firefox browser in all supported operating systems. Code repository: https://github.com/open-eid/firefox-pkcs11-loader Wiki: https://github.com/open-eid/firefox-pkcs11-loader/wiki	RIA/SK
OSX native certificate selection and PIN dialog	PIN dialog and certificate selection windows provided by the operating system's native components.	-
Minidriver	Described in chap. Drivers	RIA/SK

Drivers

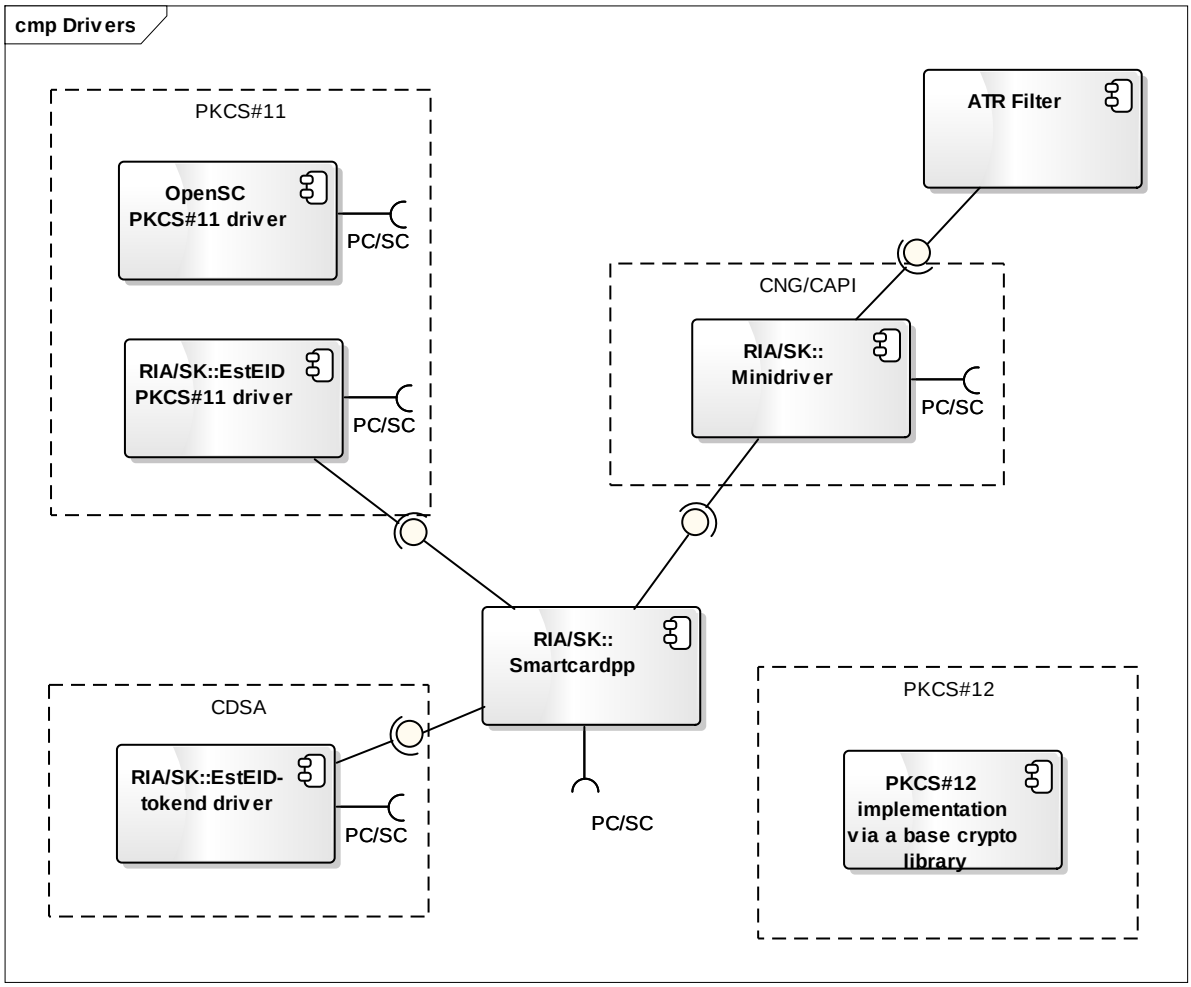

Figure 8. Cryptographic tokens' drivers

Table 8. Cryptographic token driver components

Component	Description	Owner/ Developer
EstEID PKCS#11 driver	A driver for accessing eID-cards. Connects with the card via the operating system's native PC/SC interface. Used as a default driver for signature creation with eID card in browser environment in case of OSX platform. Used as a default driver for authentication with eID card in browser environment in case of Firefox browser in OSX platform. Code repository: https://github.com/open-eid/esteid-pkcs11 Wiki: https://github.com/open-eid/esteid-pkcs11/wiki	RIA/SK
OpenSC PKCS#11 driver	A driver for accessing eID-cards. Connects with the card via the operating system's native PC/SC interface. Used as a default driver for authentication with eID card in browser environment in case of Windows and Linux platforms. Used as a default driver for signature creation in web browser environment in case of Linux platform. Wiki: https://github.com/OpenSC/OpenSC/wiki	-
Smartcardpp	eID card driver's helper component. Inner component. Code repository: https://github.com/open-eid/smartcardpp Wiki: https://github.com/open-eid/smartcardpp/wiki	RIA/SK

Component	Description	Owner/ Developer
Minidriver	Used as a default driver for accessing Estonian eID-cards via CNG interface for signature creation in web browser environment in case of Windows platform. Used as a default driver for authentication with eID card in Chrome and Internet Explorer browsers in case of Windows platform. Code repository: https://github.com/open-eid/minidriver Wiki: https://github.com/open-eid/minidriver/wiki	RIA/SK
ATR Filter	Base component for Minidriver (see http://support.microsoft.com/kb/981665 for more information).	-
Esteid Tokend	A driver for accessing eID-cards. Connects with the card via the operating system's native PC/SC interface. Used as a default driver for authentication with eID card in browser environment in case OSX platform. Code repository: https://github.com/open-eid/esteid-tokend Wiki: https://github.com/open-eid/esteid-tokend/wiki	RIA/SK
PKCS#12 implementation via base library	An implementation of PKCS#12 interface by the component's base libraries.	-

PKCS#11 driver interfaces

Components:

1. EstEID PKCS#11 driver
2. OpenSC PKCS#11 driver

Provided:

1. PKCS#11 API
    a. User: a browser signing module, software library
    b. Accessible with: C++
    c. Documentation:
        i. PKCS#11 API: http://www.emc.com/emc-plus/rsa-labs/standards-initiatives/pkcs-11-cryptographic-token-interface-standard.htm
        ii. source code for the list of implemented functions

Required:

1. PC/SC: see chap. PC/SC driver

Minidriver interfaces

Provided:

1. CNG/Minidriver API
    a. User: a browser signing module, software library
    b. Accessible with: C/C++
    c. Documentation:
        i. CNG: http://msdn.microsoft.com/en-us/library/windows/desktop/aa376210(v=vs.85).aspx,

ii. Minidriver API: http://msdn.microsoft.com/en-us/library/windows/hardware/dn631754(v=vs.85).aspx
iii. source code for the list of implemented functions

2. CAPI/Minidriver API
    a. User: a browser signing module, software library
    b. Accessible with: C/C++
    c. Documentation:
        i. CAPI: http://msdn.microsoft.com/en-us/library/aa380256.aspx
        ii. Minidriver API: http://msdn.microsoft.com/en-us/library/windows/hardware/dn631754(v=vs.85).aspx
        iii. source code for the list of implemented functions

Required:

1. PC/SC: see chap. PC/SC driver

PKCS#12 implementation via base library

Provided:

1. PKCS#12 interface
    a. User: a software library
    b. Accessible with: PKCS#12 API
    c. Documentation: see documentation of the respective component's appropriate base library

Tokend driver interfaces

Components implementing the interface:

1. EstEID Tokend driver

Provided:

1. CDSA
    a. User: software library
    b. Accessible with: C++
    c. Documentation: see https://developer.apple.com/library/mac/documentation/security/conceptual/cryptoservices/CDSA/CDSA.html

Required:

1. PC/SC: see chap. PC/SC driver

PC/SC driver interfaces

Provided:

1. PC/SC interface
    a. User: eID-card's driver

b. Accessible with: PC/SC API

c. Documentation: see http://www.pcscworkgroup.com/specifications/overview.php

Required: not in the scope of this document.

Updating mechanisms

The following chapter describes automatic updating mechanisms of different ID-software desktop applications. Several combinations of central software update checking and distribution environments are used depending on the end-user's operating system.

Windows updating mechanism

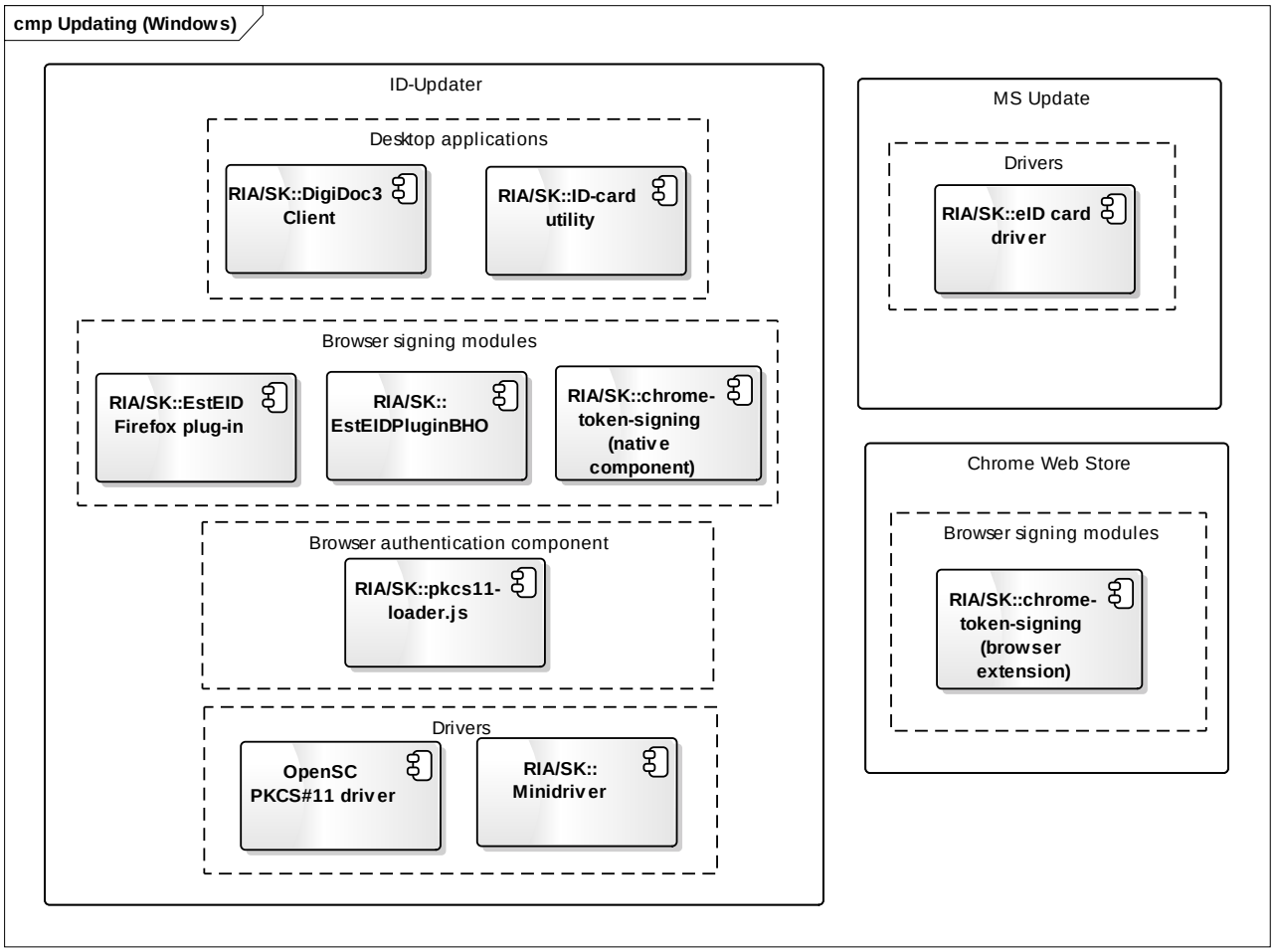

Figure 9. Updating mechanisms in Windows

Table 9. Updating mechanisms in Windows

Component	Description	Owner/ Developer
ID-updater	Service that is periodically checks if newer versions of related ID-software components are available for download, initiates the download and installation if necessary.	RIA/SK
MS Update	Microsoft Update – see Microsoft's documentation for more information.	-
Chrome Web Store	See https://chrome.google.com/webstore/detail/token-signing/ckjefchnfjhjfedoccjbhjpbncimppeg	-

OS X updating mechanism

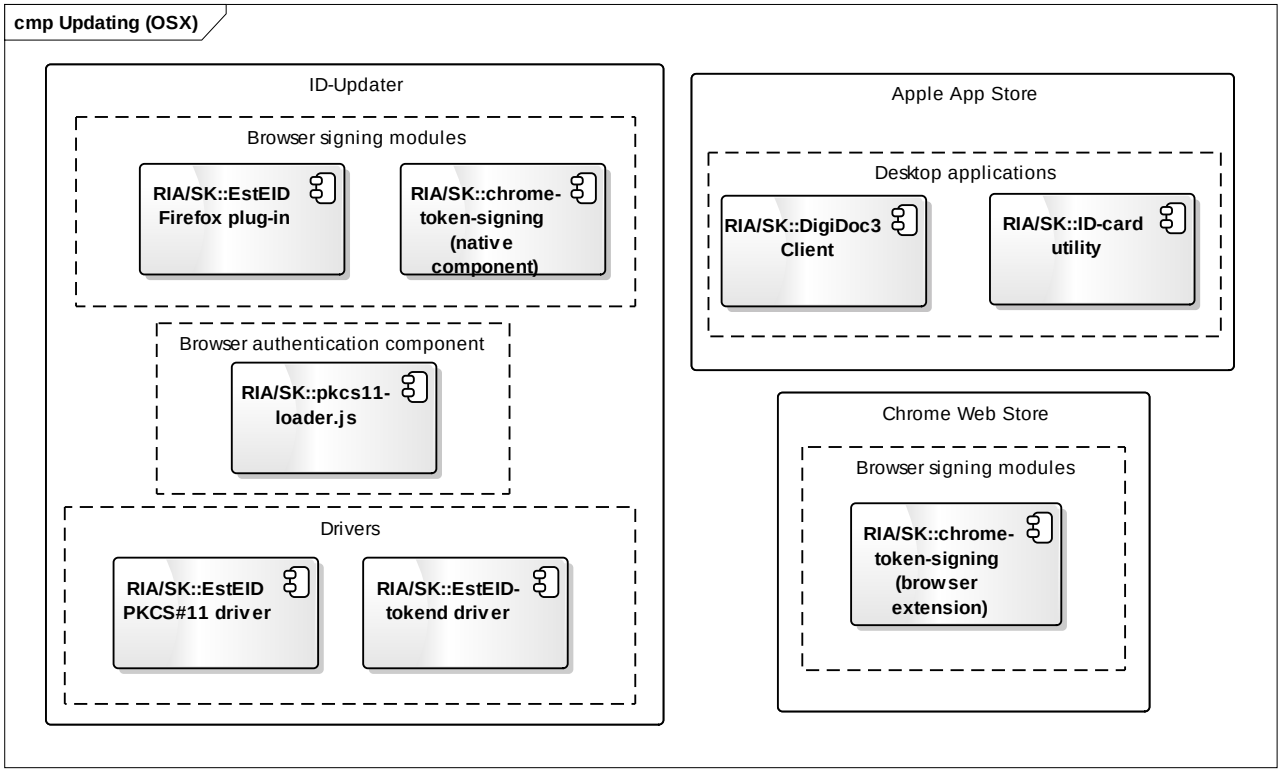

Figure 10. Updating mechanisms in OSX

Table 10. Updating mechanisms in OSX

Component	Description	Owner/ Developer
ID-updater	Described in chap. Windows updating mechanism	RIA/SK
Apple App Store	See Apple App Store documentation.	-
Chrome Web Store	See https://chrome.google.com/webstore/detail/token-signing/ckjefchnfjhjfedoccjbhjpbncimppeg	-

Linux updating mechanism

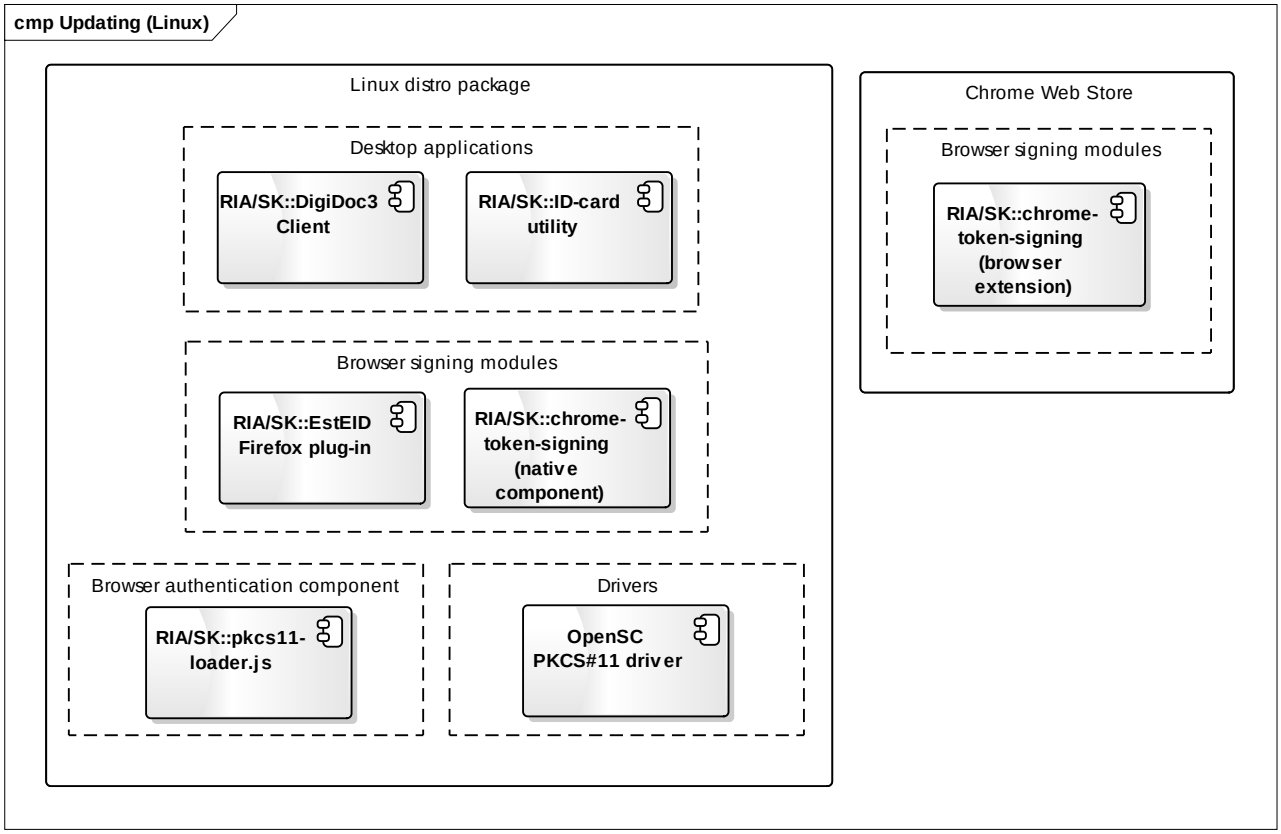

Figure 11. Updating mechanism in Linux

Table 11. Updating mechanisms in OSX

Component	Description	Owner/ Developer
Ubuntu package updates	Managed and maintained by SK. The binary packages are released for installation and updating to https://installer.id.ee/media/ubuntu/ repository.	RIA/SK
Packages updates for other distros	Managed by the open-source community. Packages are built, added and updated in Estobuntu and Fedora distributions by the package maintainers.	-
Chrome Web Store	See https://chrome.google.com/webstore/detail/token-signing/ckjefchnfjhjfedoccjbhjpbncimppeg	-

External services' interfaces

ID-updater interface

Interface's properties:

1. User: DigiDoc3 Client
2. Accessible with: HTTPS protocol
3. Accessible from:
    a. Windows: https://installer.id.ee/media/win/products.xml,
    b. OSX: https://installer.id.ee/media/osx/products.xml,
    c. Linux: https://installer.id.ee/media/ubuntu/pool/main/

Kill switch service interface

Interface's properties:

1. User: DigiDoc3 Client
2. Accessible with: XML file sent over HTTPS protocol
3. Accessible from: https://installer.id.ee/media/killswitch/products.xml

DigiDocService web service interface

Interface's properties:

1. User: DigiDoc3 Client
2. Accessible with: SOAP 1.0-encoded over HTTPS
3. Accessible from: https://digidocservice.sk.ee
4. Documentation: http://www.sk.ee/upload/files/DigiDocService_spec_eng.pdf

Error reports repository interface

Interface's properties:

1. User: DigiDoc3 Client, ID-utility
2. Accessible with: HTTPS protocol
3. Accessible from: https://cr.eesti.ee

LDAP directory interface

Interface's properties:

1. User: DigiDoc3 Client
2. Accessible with: LDAP protocol
3. Accessible from: ldap.sk.ee:389
4. Documentation: https://sk.ee/en/repository/ldap/ldap-kataloogi-kasutamine/

TSL repositories' interfaces

Interface's properties:

1. User: Libdigidocpp, DigiDoc4j
2. Accessible with: HTTPS protocol
3. Accessible from:
    a. European Commission's master list: https://ec.europa.eu/information_society/policy/esignature/trusted-list/tl-mp.xml
    b. National TSL URLs in the European Commission's TSL, e,g, Estonian TSL: http://sr.riik.ee/tsl/estonian-tsl.xml
4. Documentation: http://www.etsi.org/deliver/etsi_ts/119600_119699/119612/01.02.01_60/ts_119612v010201p.pdf

Time-stamping service interface

Interface's properties:

1. User: Libdigidocpp , DigiDoc4j
2. Accessible with: HTTPS protocol
3. Accessible from:
    a. SK's time-stamping service http://demo.sk.ee/tsa/
4. Documentation: RFC3161

OCSP service interface

Interface's properties:

1. User: DigiDoc4j, JDigiDoc, Libdigidocpp, CDigiDoc software libraries; DigiDocService web service
2. Accessible with: HTTPS protocol
3. Accessible from:
    a. SK's OCSP service for SK issued certificates: http://ocsp.sk.ee/
    b. SK's Proxy OCSP service for international use: http://ocsp.sk.ee/_proxy
    c. SK's test OCSP service: http://demo.sk.ee/ocsp
4. Documentation: RFC6960

ID-card owners' photo repository interface

Interface's properties:

1. User: ID-card utility program
2. Accessible with: HTTPS protocol

3. Accessible from: https://sisene.www.eesti.ee/idportaal/portaal.idpilt

Certificate renewal service interface

Interface's properties:

1. User: ID-card utility program
2. Accessible with: HTTP
3. Accessible from: http://www.sk.ee:80/id-kontroll2/usk/

Eesti.ee e-mail checking service interface

Interface's properties:

1. User: ID-card utility program
2. Accessible with: HTTPS
3. Accessible from: https://sisene.www.eesti.ee/idportaal/postisysteem.naita_suunamised

Mobile-ID validity checking service interface

Interface's properties:

1. User: ID-card utility program
2. Accessible with: HTTPS
3. Accessible from: https://id.sk.ee/MIDInfoWS/

Deployment model

The following subchapters describe physical deployment of ID-software components in collaboration with external components that were depicted in chap. Component model in case of the most common use cases.

Signing in web browser

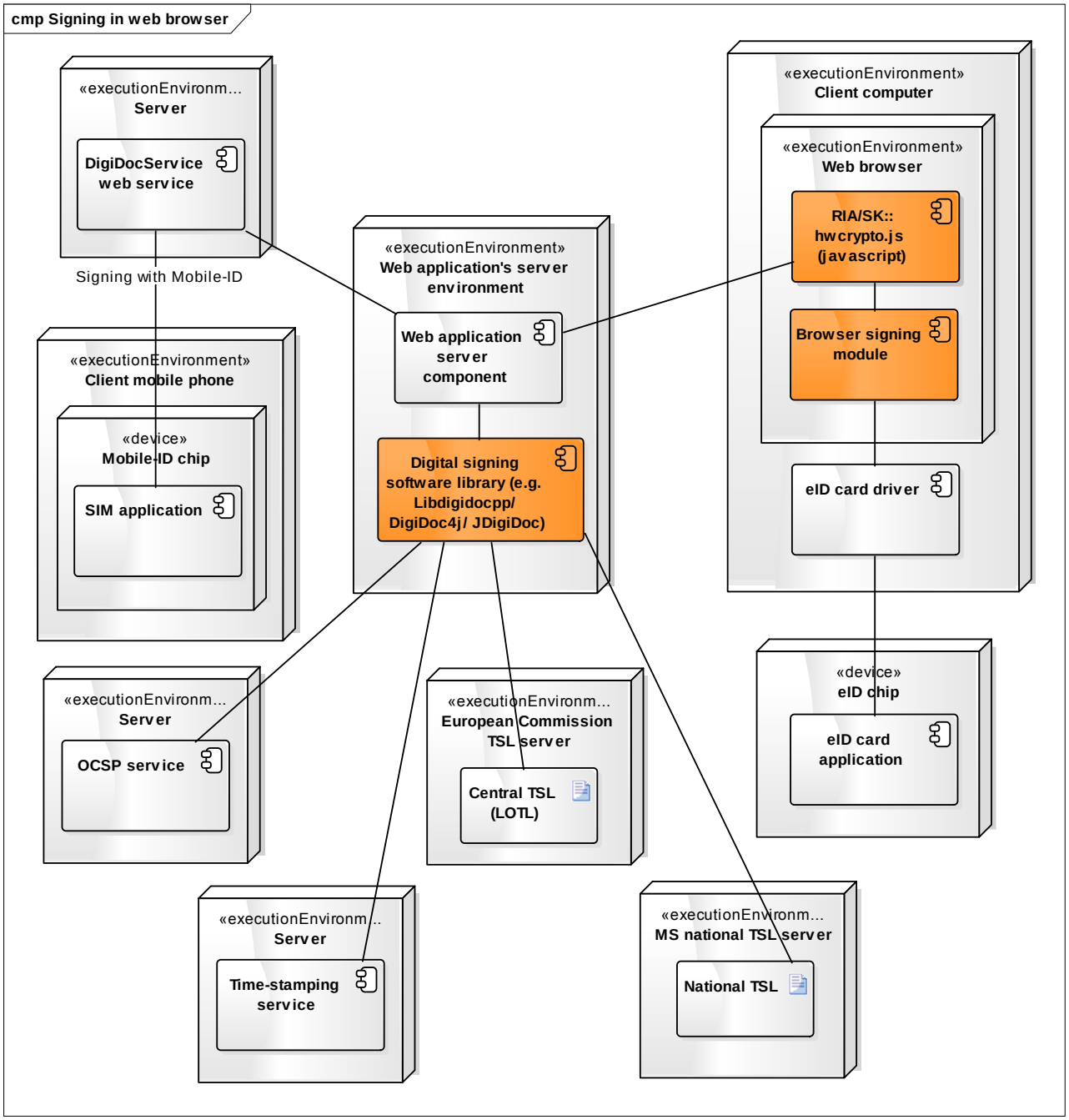

Figure 12. Signing in web browser via a web application

Additional notes:

- A DigiDoc software library (i.e. DigiDoc4j, JDigiDoc, Libdigidocpp or CDigiDoc library) and DigiDocService web service are optional and can be used for creating a DDOC or BDOC container and adding the created signature value to the container.
- Long term validation data is added to the DDOC/BDOC signature by obtaining OCSP confirmation and optionally a time-stamp

- DigiDocService is required in order to sign with Mobile-ID.
- Signature value is calculated either in the Mobile-ID SIM card or eID-card's chip.
- When signing with eID smartcard then the browser signing module is necessary for enabling communication with the smart card connected to the user's system. Hwcrypto.js library offers a single API for supporting signing modules of all the supported browsers.
- Optionally, trust anchor data is retrieved from TSL lists – the European Commission's central TSL and national TSL's of the EU member states.

Signing with DigiDoc3 Client

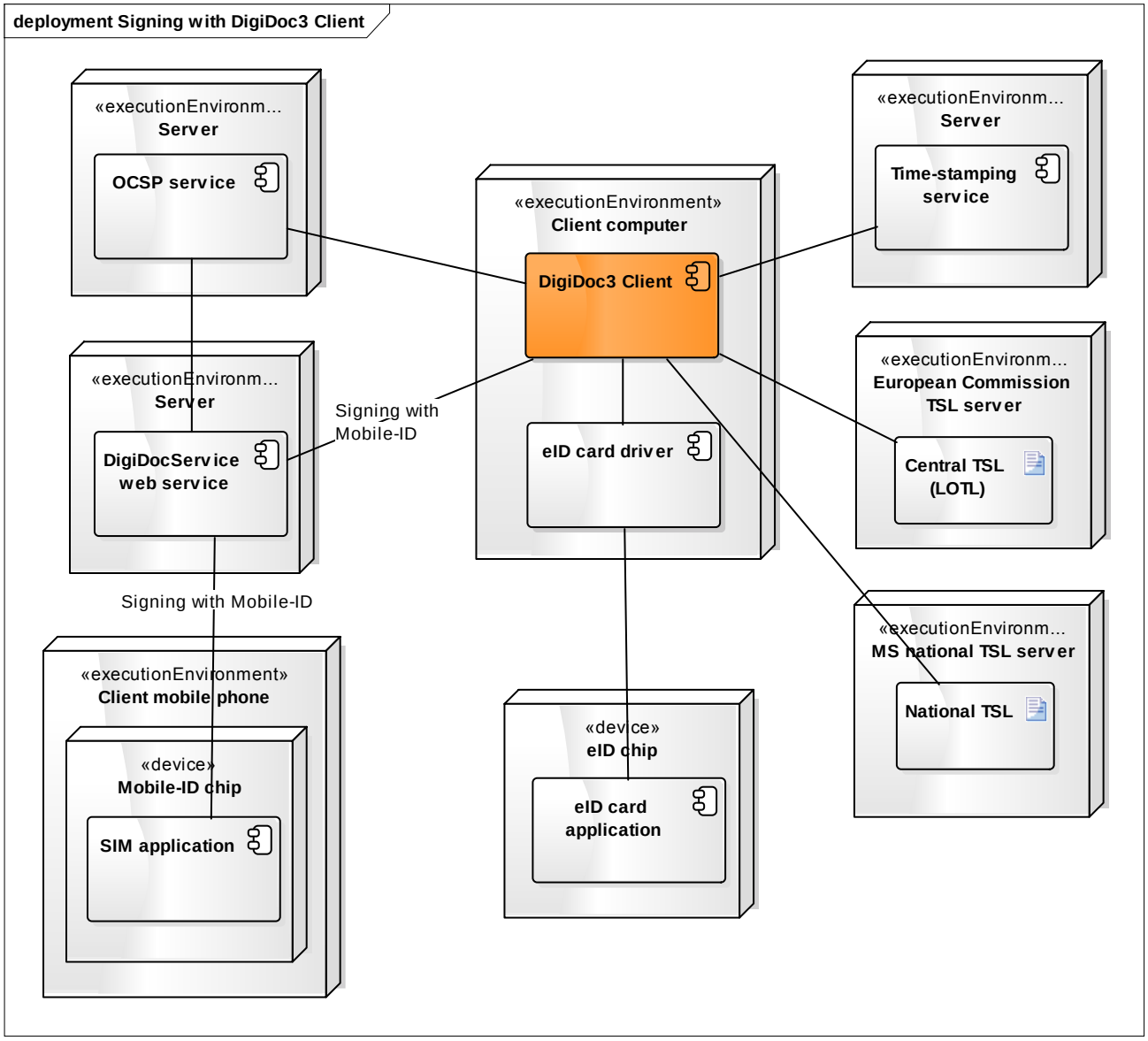

Figure 13. Deployment of components during signature creation with DigiDoc3 Client

Additional notes:

- DigiDoc3 Client is used for creating the DDOC or BDOC (XAdES/ASiC-E) container and adding the signature value to the container.
- Long term validation data is added to the DDOC/BDOC signature by obtaining OCSP confirmation and optionally a time-stamp
- DigiDocService is required in order to sign with Mobile-ID.
- Signature value is calculated either in the Mobile-ID SIM card or ID-card's chip.
- Trust anchor data is retrieved from TSL lists – the European Commission's central TSL and national TSL's of the EU member states.