BRZ
PDF-AS
API Spezifikation
R 1.0
19.11.2007

Final

NETCONOMY Onlineberatungs- & -management GmbH & Co KG
Lange Gasse 42 / Körösistrasse 38, A-8010 Graz
T ++43(0) 316/ 815544, F ++43(0) 316 / 815544 – 99
www.netconomy.net, office@netconomy.net
Firmenbuch : FN 205042h, UID : ATU 50921800

Inhaltsverzeichnis

1 Einleitung

1.1 Zweck dieses Dokumentes

Dieses Dokument beschreibt die im Zuge des Projekts „Unterstützung bei der Integration von PDF-AS für Signatur- und Signaturprüfungsservices“ (Angebot vom 30.10.2007) umzusetzenden Änderungen an der PDF-AS Applikation auf technischer Ebene sowie einleitend die Hintergründe, welche zu den diversen technischen Entscheidungen geführt haben.

Hauptziel der Änderungen ist die Schaffung einer technisch klar definierten Java Schnittstelle (API) zwischen BRZ und der PDF-AS Applikation, die es ermöglicht PDF-AS Funktionalität aus Java Applikationen heraus zu nutzen.

1.2 Gültigkeit dieses Dokuments

Als nicht finale Version ist dieses Dokument als Entwurf und Diskussionsgrundlage zu betrachten. Es hat keinen bindenden Charakter.

Als finale Version ist dieses Dokument vom Auftraggeber gesondert abzunehmen.

Dieses Dokument wurde gemeinsam mit dem Auftraggeber abgestimmt und beinhaltet eine detaillierte Spezifikation aller Funktionalitäten aufgrund der Anforderungen entsprechend dem Angebot bezüglich der API. Der Auftraggeber hat sich von der Vollständigkeit dieses Dokumentes überzeugt.

1.3 Zusammenhang mit anderen Projektphasen

Dieses Dokument bezieht sich ausschließlich auf die Projektphase zur Umsetzung der API und nicht auf die Integrationsphase und die Prüfergebnisfilter.

1.4 Dokumenthistorie

Diese Dokumenthistorie listet nur jene Versionen, die an den Auftraggeber übermittelt wurden:

Version	Autor	Datum	Beschreibung
0.3	Prinz	12.11.2007	
1.0	Prinz	19.11.2007	

1.5 Anhänge

Die folgende Tabelle listet alle Anhänge zu diesem Dokument auf, die als Teil desselben zu betrachten sind:

Anhang	Version	Datum	Dokumentname
I	1.0	19.11.2007	2007_11_19-BRZ_PDF-AS_API_JavaSrc_R1.0_wp.zip

2 PDF-AS

Die PDF Amtssignatur (PDF-AS) Applikation ermöglicht es, PDF- Dokumente mit Amtssignaturen zu versehen und diese auch zu prüfen. PDF-AS stellt daher folgende zwei Funktionalitäten bereit:

- Aufbringen einer Amtssignatur auf ein PDF Dokument
- Korrektheitsprüfung einer solchen Amtssignatur auf einem PDF-Dokument

Die eigentliche kryptografische Signaturerstellung bzw. Signaturprüfung wird nicht von PDF-AS, sondern von einem externen Signaturerstellungsgerät bzw. Signaturprüfgerät durchgeführt. Gegenwärtig unterstützt PDF-AS folgende zwei Geräte:

- die Bürgerkartenumgebung (BKU)
- das Modul für Online Anwendungen Signaturprüfung und Server Signatur (MOA SPSS)

Die eigentliche Aufgabe von PDF-AS ist es, die zu signierenden Daten aus einem gegebenen PDF Dokument zu extrahieren um diese in Folge vom Signaturerstellungsgerät signieren zu lassen und diese Signaturinformation in Form einer Amtssignatur auf das PDF Dokument aufzubringen. Entsprechend der Natur der signierten Daten werden folgende zwei Signaturtypen unterschieden:

- Textsignatur
- Binärsignatur

Bei der Signaturprüfung mittels PDF-AS werden aus einem gegebenen PDF Dokument alle Amtssignaturen extrahiert und deren Signaturdaten dann vom entsprechenden Signaturprüfgerät geprüft.

Die PDF-Applikation kann auf zwei verschiedene Arten bedient werden:

- über die Kommandozeile
- über eine Web Applikation

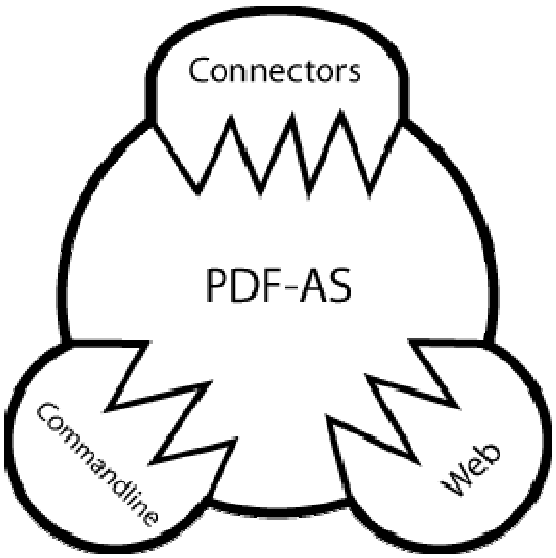
2.1 Komponenten innerhalb von PDF-AS

Entsprechend der oben genannten Anwendungsfälle kann die PDF-AS Applikation in folgende Komponenten gegliedert werden:

- Kern
  Der Kern der PDF-AS Applikation verarbeitet PDF Dokumente und implementiert die Algorithmen, welche im Wesentlichen die für eine Signatur bzw. Prüfung benötigten Daten (binär oder textuell) zusammenstellen.
- Connectoren
  Die Connectoren übernehmen die eigentliche Kommunikation mit den Signaturerstellungs- bzw. Prüfgeräten. Sie senden die Daten an das entsprechende Gerät und werten dessen Antwort aus, welche in weiterer Folge wieder vom Kern verarbeitet wird.
- Commandline
  Die nötigen Applikationsparameter werden über die Kommandozeile eingegeben.
- Web
  Die nötigen Applikationsparameter werden über eine HTML Oberfläche eingegeben.

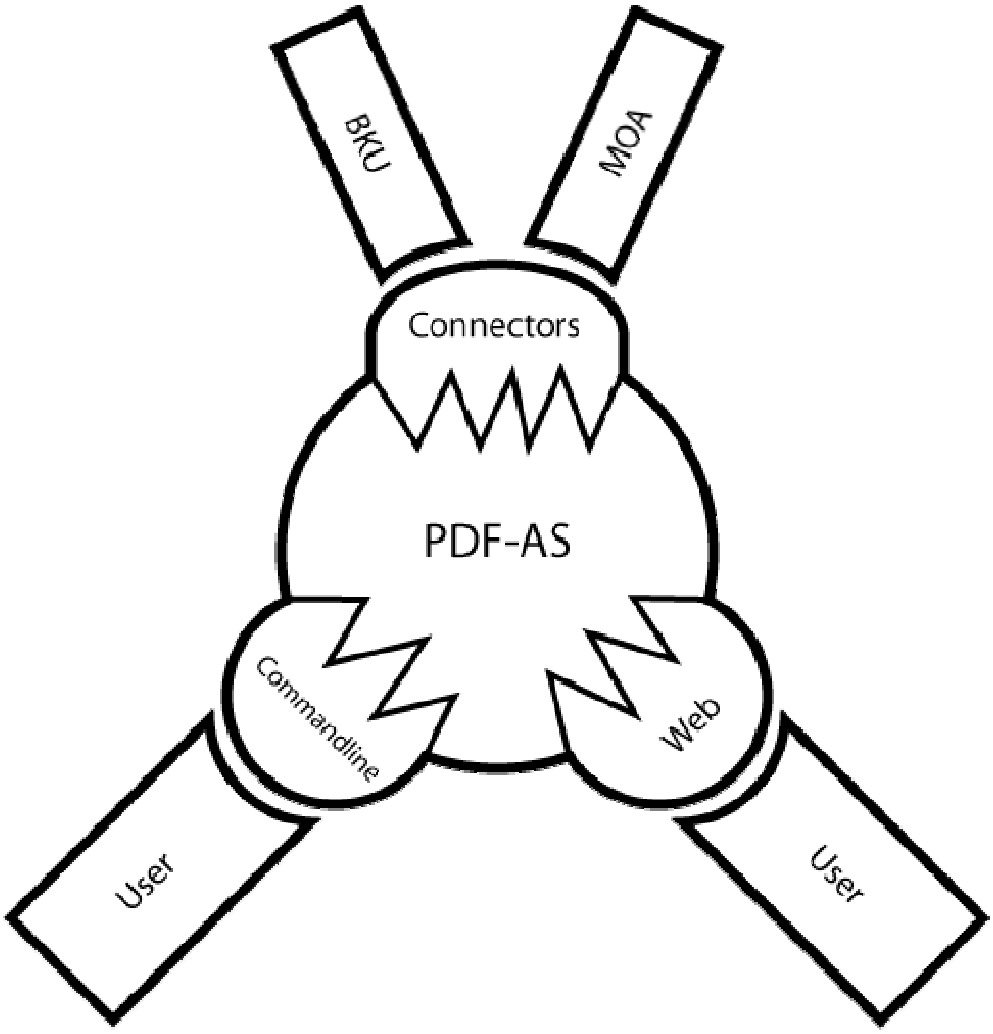
2.2 Ist Zustand

Gegenwärtig existiert keine klare scharfe Trennung der einzelnen Komponenten (Kern, Connectors, Commandline, Web) innerhalb der PDF-AS Applikation – sie sind Stark ineinander verflochten. Folgende Grafik illustriert den internen Aufbau von PDF-AS.



Diese Situation ist teilweise historisch bedingt und resultiert noch aus den Anfangszeiten des PDF-AS Projekts. PDF-AS wurde ursprünglich als eine einzige große Applikation entwickelt, welche möglichst einfach vom User bedient werden hätte können sollen.

Die User-zentrierten Hauptanwendungsfälle waren „Zugriff über Kommandozeile“ und „Zugriff über eine Weboberfläche“. Da die PDF-AS Applikation selbst nicht als Signaturapplikation gedacht ist kommuniziert sie mit solchen (MOA/BKU) über deren HTTP Schnittstellen. In folgender Grafik sind der PDF-AS Kern sowie die gegenwärtigen Kommunikationsschnittstellen eingezeichnet.



Die starke Verzahnung der PDF-AS Komponenten untereinander stellt ein Problem für jene Szenarien dar, in welchen die gesamte Funktionalität von PDF-AS, oder nur Teile davon, in eine bereits bestehenden (Java) Applikation integriert werden sollen. Gegenwärtig müsste man dazu den entsprechenden Teil (z.B. Commandline) herauslösen und/oder erweitern. Im jedem Fall ist dazu ein direkter Eingriff in den PDF-AS Quellcode nötig.

Um diese Situation zu entschärfen ist es notwendig die internen Verstrickungen aufzulösen und nach außen hin eine klare Schnittstelle zu definieren.

3 BRZ

Das Bundesrechenzentrum (BRZ) möchte die PDF-AS Funktionalität nutzen. Es besteht bereits ein beim BRZ eingesetztes TIBCO Framework, in welches die PDF-AS Funktionalität eingebettet werden soll. Als Signaturerstellungs- bzw. Prüfgerät wird MOA SP/SS.

3.1 Technische Anforderungen

Im Meeting am 8. 11. 2007 wurden für den Einsatz von PDF-AS im Rahmen des BRZ folgende Anforderungen identifiziert:

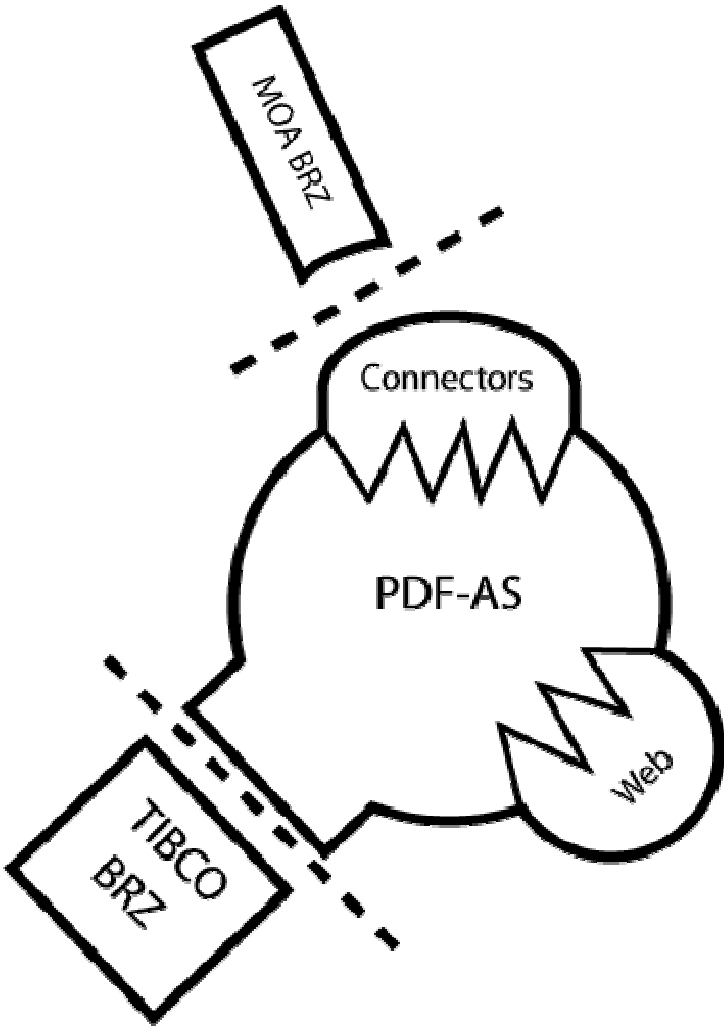
- Das bereits bestehende Framework TIBCO ruft die PDF-AS Funktionalität über das Java API auf.
- PDF-AS und die benötigten Libraries (PDF-AS Modifikation von iText, PDF-AS Modifikation von pdfBox, …) werden als JAR in die TIBCO Umgebung eingebunden.
- Die eigentliche Signatur und Prüfung wird von MOA SS bzw. MOA SP durchgeführt, welches über den entsprechenden Connector (Detached, HTTPS) von PDF-AS angesteuert wird.
- Die BKU wird nicht verwendet.
- TIBCO und MOA sind in der BRZ Infrastruktur installiert, konfiguriert und werden vom BRZ gewartet.
- Die abgestimmten Signaturprofile (Bildmarke, Text, …) des BRZ werden in der PDF-AS Config hinterlegt.
- Es gibt mehrere solcher Profile und es können zu späteren Zeitpunkten neue Profile hinzukommen. D.h. die Config muss „neu ladbar“ sein.
- Der MOA KeyIdentifier muss über das API aus der Konfiguration auslesbar sein. Sinn dahinter ist, dass TIBCO dem MOA KeyIdentifier ein entsprechendes MOA Profil zuordnen muss.
- Es werden beide Signaturtypen, binär und textuell, über das API angeboten. Aus Performance Sicht ist in jedem Fall die Binärsignatur der Textsignatur vorzuziehen. Allerdings ist nur die Textsignatur von Papier (theoretisch) rückführbar.
- Für die PDF-AS Applikation muss ein Work Directory zur Verfügung stehen, welches den PDF-AS Anforderungen entspricht. Die wichtigsten Elemente des Work Directories seien hier aufgeführt:
    - Config mit Signaturprofilen und Verbindungsparametern
    - Connector Templates
    - „Swap Space“ auf welche Zwischenergebnisse im Dateisystem abgelegt werden können. D.h. es sind für das Work Directory Lese- und Schreibrechte erforderlich.
    - Local Certificate Store
    - Weitere Punkte für das Work Directory siehe PDF-AS Dokumentation.
- Als Ziel Java Version wurde vom BRZ Java 1.5 genannt. Da PDF-AS vollständig in Java 1.4 verfasst ist erfüllt es diese Voraussetzung implizit.
- Die PDF-AS Web Applikation wird für das BRZ Projekt nicht gebraucht.

3.2 Auswirkungen

Die oben genannten Anforderungen, speziell die nicht Erforderlichkeit der Web Applikation, haben auf die Konzeption des PDF-AS API im Sinne von BRZ folgende Auswirkungen:

- Der Web Applikations Teil wird auf Grund seiner aufwändigen Asynchronitäten nicht in das API aufgenommen.
- Ebenso werden die Connectoren nicht in das API aufgenommen.
- Das API bezieht sich stark auf den Commandline Teil. Die Commandline Komponente wird also zum ersten „Nutzer“ des API.

Folgende Grafik beschreibt die Änderungen an der PDF-AS Applikation im Zuge des BRZ Projektes:



4 PDF-AS Abstraktion

Das folgende Kapitel beschreibt die Abstraktionssicht auf die PDF-AS Applikation („Black Box“). Es werden hierbei die für das API Aufrufe notwendigen Ein- und Ausgabeparameter aufgezeigt.

4.1 PDF-AS Umgebung

PDF-AS benötigt eine Arbeitsumgebung (Work directory). Bevor die Funktionalität des APIs (Signatur, Prüfung) genutzt werden kann muss diese Umgebung eingerichtet werden. Das Work Directory enthält unter anderem das Config File sowie den Local Certificate Store (siehe PDF-AS Dokumentation).

Beim Anlegen einer API Instanz wird PDF-AS das Work Directory mitgeteilt. API Aufrufe arbeiten dann in dieser Umgebung.

4.2 Signatur

Eingabe:

- PDF Dokument
- Signaturtyp
  binär, textuell
- Signaturerstellungsgerät
  BKU, MOA
- Signaturprofil Identifier
  (optional)
  Entsprechend dem übergebenen Profil Identifier muss im Config ein Profil angelegt sein.
  Wird kein Profil Identifier übergeben, so wird das im Config angegebene Default Profil ausgewählt.
- Signaturposition
  (optional)
  analog zum Kommandozeilenparameter siehe PDF-AS Dokumentation.
  Es sei darauf hingewiesen, dass in Punkto Performance eine absolute Positionierung einer automatischen Positionierung vorzuziehen ist.

Ausgabe:

- Signiertes PDF Dokument
- Signaturzertifikat
- Signaturposition
  Es wird die im Endeffekt gewählte Position zurückgegeben. Dies kann vor allem bei automatischer Positionierung für nachfolgende Operationen interessant sein.

Umgebung:

- Signaturprofil im Config definiert
- Signaturerstellungsgeräteparameter im Config definiert
- Signaturposition eventuell im Config File definiert
  Dies wird nur verwendet wenn keine Angabe eines Positionierungs Eingabeparameters erfolgt.

4.3 Prüfung

Eingabe:

- PDF Dokument
- Signaturprüfgerät
  BKU, MOA
- Prüfungs-Modus
  (optional)
  Mitgeben, ob ausschließlich binär oder auch Textuell geprüft werden soll. Eine ausschließlich binäre Prüfung ist aus
- Zu prüfende Signatur
  (optional)
  Gibt an, welche der gefundenen Signaturen geprüft werden soll.
  Das Standardverhalten ist alle zu prüfen.
- Verifikationszeitpunkt
  (optional)
  Gibt den Zeitpunkt an, zu welchem die Prüfung durchgeführt werden soll.
  Wird zum Signaturprüfgerät durchgeschliffen.

Ausgabe:

- Liste mit Prüfergebnissen:
  - Signaturtyp
    Typ der Signatur (textuell, binär).
  - Signaturdaten
  - Signaturzertifikat
    Liste mit den Signaturzertifikaten entsprechend den Prüfergebnissen.
  - Signierzeitpunkt
  - Extrahierte Signaturprofilinformation
    Signaturprofilinformation (falls vorhanden)
  - Prüfergebnis(se)
    Prüfergebnisse (Zertifikats-Check, Signatur-Check, Manifest-Check).
  - Qualifizierung des Zertifikats
    Attribut, ob das Zertifikat „qualified“ ist.
  - Verifikationszeitpunkt

Umgebung:

- Alle potentiellen Signaturprofile im Config definiert (notwendig für das Finden von Textsignaturen)

4.4 Analyse und Prüfung

Der Prozess der Prüfung kann ebenso in zwei Schritten durchgeführt werden:

- Analyse:
  Das gegebene PDF Dokument wird analysiert und alle Signaturen ermittelt.
- Prüfung:
  Die aus der Analyse stammenden Daten werden geprüft.

Analyse-Eingabe:

- PDF Dokument
- Prüfungs-Modus
  siehe oben

Analyse-Ausgabe:

- Liste mit gefundenen Signaturinformationen:
  - Signaturtyp
  - Signaturdaten
  - Signaturzertifikat
  - Signierzeitpunkt
  - Extrahierte Signaturprofilinformation

Prüfung-Eingabe:

- Analyse-Ausgabe
- Signaturprüfgerät
  siehe oben
- Verifikationszeitpunkt
  siehe oben

Prüfung-Ausgabe:

- Siehe oben

Umgebung:

- Siehe oben

4.5 Config Reload

Dieser API Aufruf macht Änderungen am Config File für die gegenwärtige Umgebung wirksam.

Normalerweise ist eine Änderung am ConfigFile notwendig, wenn ein neues Signaturprofil eingeführt wird.

4.6 Profilinformation abrufen

Für die gegenwärtige Umgebung und das gegenwärtige Config File wird eine Liste mit den Profilinformationen aller vorhandenen Profile angezeigt.

Die Profilinformation eines Profils beinhaltet:

- Profile Identifier
- MOA KeyIdentifier

5 PDF-AS Java API

Auf eine detaillierte Auflistung der Java API Klassen und Methoden sowie deren Parameter wird hier verzichtet und auf das entsprechende JavaDoc verwiesen. Hier wird nur ein kurzer Überblick über das API gegeben.

5.1 Zusammenfassung des API

Folgende Punkte charakterisieren das Java API:

- Zentrales Element ist das PDF-AS Objekt, welches das PDF-AS Interface implementiert.
  - Das PDF-AS Objekt wird per Factory Methode angelegt.
  - Beim Erstellen des Objekts mittels Factory Methode wird das Work Directory als Parameter angegeben.
  - Es wird beim Anlegen des PDF-AS Objekts ein implizites Config Reload durchgeführt. D.h. das PDF-AS Objekt ist nach dem Anlegen bereits einsatzbereit.
- Das PDF-AS Interface stellt oben beschriebene Methoden bereit. Für Details siehe JavaDoc.
- Die umfangreichen Parameter an die Methoden werden über Parameter Objekte übergeben.