

MOA-ZS Technische Spezifikation 9. April 2004	Spezifikation
	MZSSPEC
	Entwurf öffentlich

Bezeichnung	MOA-ZS– Technische Spezifikation
Kurzbezeichnung	MZSSPEC
Version	Version 1.0
Dokumentklasse	Spezifikation
Dokumentstadium	Entwurf öffentlich
Kurzbeschreibung	Diese Spezifikation beschreibt die Abläufe innerhalb MOA-ZS und die Datenübergabe an MOA-ZS
Zielgruppe	Applikationsentwickler die MOA-ZS nutzen wollen MOA-ZS Entwickler
Autor	Larissa Naber larissa.naber@cio.gv.at Michael Liehmann michael.liehmann@cio.gv.at
Editor	Arno Hollosi arno.hollosi@cio.gv.at Peter Reichstädter peter.reichstaedter@cio.gv.at
Arbeitsgruppe	Elektronische Zustellung Stabstelle IKT Strategie Bundes



Inhaltsverzeichnis

1	Einleitung	4
2	Zustellstückannahme	6
2.1	Funktionaler Überblick	6
2.2	Datenformat für die Dokumentenanolieferung (DeliveryRequest) .	9
2.2.1	Applikation (Sender)	10
2.2.2	Empfänger (Receiver)	10
2.2.3	Zustell-Metainformation	12
2.2.4	Zustellstück – Payload	12
2.3	Datenformat für Erfolgs- und Fehlermeldungen in der Dokument- anolieferung (DeliveryResponse)	13
2.4	Datenformate der Benachrichtigungen (DeliveryNotification)	17
2.5	Vollständigkeitsprüfung	17
3	Prüfung der Adressierbarkeit	19
3.1	bPK Umrechnung beim SZR	19
3.1.1	Funktionaler Überblick	19
3.1.2	MOA-ZS Anfrage	19
3.1.3	Antwort des Stammzahlenregisters	21
3.2	Adressierbarkeitsanfrage beim Zustellkopf	22
3.3	Callback-Attachments	22
4	Anbringen der Absendersignatur	23
4.1	Funktionaler Überblick	23
4.2	Datenaufbereitung	24
4.2.1	XML Nachrichten	24
4.2.2	Nicht XML Zustellstücke	24
4.2.3	XML und Nicht XML Dateien in Kombination	25
4.3	MOA-SS Interface: Request	25
4.3.1	CreateXMLSignaturRequest	25
4.3.2	CreateSignatureInfo	26
4.3.3	DataObjectInfo	27
4.4	MOA-SS Interface: Response	30

<i>MOA-ZS: Technische Spezifikation Version 1.0</i>	3
5 Verschlüsselung des Zustellstücks	32
5.1 Zustellstückaufbereitung	32
5.2 Verschlüsselung	36
6 Übergabe an einen Zustelldienst	37
7 Konfiguration	38
7.1 Applikationen	38
7.2 Komponenten	39
7.3 MOA-ZS Interne Konfiguration	39
8 Logging	40
A Beispiele	41
A.1 Beispiel: Anlieferung eines Zustellstücks	41
A.2 Beispiel: Mailbody	42

Kapitel 1

Einleitung

MOA-ZS ist eine Middleware deren Aufgabe darin besteht Fachapplikationen den Zugang zur elektronischen Zustellung zu erleichtern in dem sie die Zahl der Interaktionen mit Modulen von 4 auf 1 reduziert. Das Hauptaugenmerk liegt auf der einfachen Handhabung der Kommunikation mit MOA-ZS. Aus diesem Grund gibt es nur eine geringe Anzahl von Konfigurationsparametern (siehe Kapitel 7) und weder Bulkabfragen noch Untermodulaufrufe.

MOA-ZS besteht aus fünf eindeutig abgegrenzten Blöcken. Diese Blöcke erledigen einzelne (optionale) Funktionalitäten, zumeist durch den Aufruf von anderen externen Anwendungen. Siehe Abbildung 1.1.

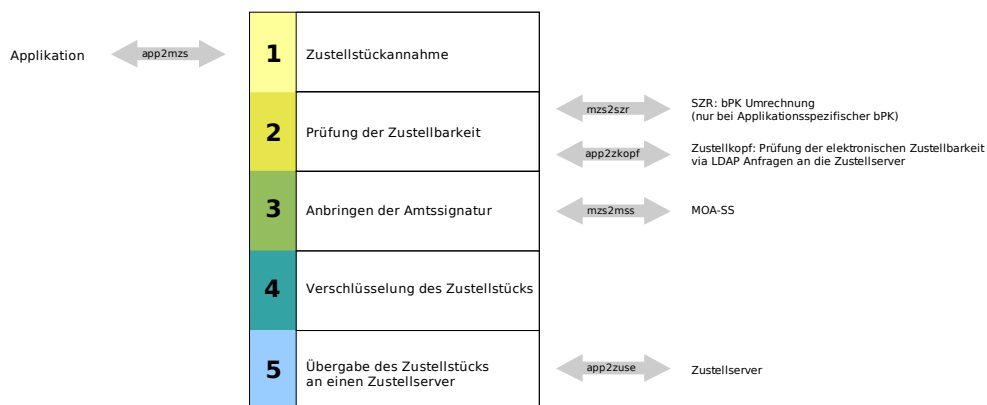


Abbildung 1.1: Überblicksdarstellung des MOA-ZS Workflows.

Die vorliegende Spezifikation behandelt nur den Ablauf innerhalb von MOA-ZS und die Schnittstelle zur Datenübernahme, sowie die Schnittstellen zum Stammszahlregister (SZR) und MOA-SS soweit sie für die Implementierung von MOA-ZS relevant sind. Der Datenaustausch mit anderen externen Anwendungen wird in den jeweiligen Interface Spezifikationen beschrieben. Das XML-Schema `app2mzs.xsd` ist ein normativer Teil dieser Spezifikation. Einen (nichtnormativen) Überblick

über die elektronische Zustellung und das Zustellungseinbindungsmodul MOA-ZS bietet die Einführung zur elektronischen Zustellung [19].

In den kommenden Kapiteln befinden sich ein Vielzahl von unterschiedlichen Namespaces. Anbei eine kurze Aufstellung: 20

Präfix	Erläuterung	Namespace
dsig	XMLDsig [1]	http://www.w3.org/2000/09/xmldsig#
soap	SOAP [22]	http://www.w3.org/2001/12/soap
saml	SAML Assertion [16]	urn:oasis:names:tc:SAML:1.0:assertion
p	PersonData 2.0 [8]	http://reference.e-government.gv.at/namespace/20020228#
zs	ZUSEMessage [12]	http://reference.e-government.gv.at/namespace/zuse11#
mzs	MOAZS Schnittstelle	http://reference.e-government.gv.at/namespace/moazs10/app2mzs#

Kapitel 2

Zustellstückannahme

2.1 Funktionaler Überblick

Die Übernahme eines Zustellstück erfolgt gemäß des Applikation \leftrightarrow MOA-ZS Schnittstelle zur Datenanlieferung (siehe Abschnitt 2.2). Alle verwendeten XML Nachrichten sind gemäß dem XML Schema app2mzs [10] zu formulieren. Im Falle von Überlast oder Ausfall von nachgegliederten (externen) Anwendungen kann ein Zustellstück auch abgewiesen werden um einen Backlog von nicht zustellbaren Dokumenten zu vermeiden. Abweisungskriterien sind konfigurierbar zu gestalten (siehe Abschnitt 7.3).

MOA-ZS übernimmt das Zustellstück von der aufrufenden Applikation. Das dabei zu verwendende Datenformat ist MOA-ZS Message **DeliveryRequest** (siehe 2.2). Bei dem Zustellstück handelt es sich um eine SOAP Message[22].

Das Zustellstück kann aus einem oder mehreren XML oder binären Dokumenten bestehen. Die Zustellmetainformation wird im XML Format innerhalb der SOAP Message übermittelt. XML Dokumente werden ebenfalls innerhalb der SOAP Message übermittelt, alle anderen Dokumente werden entweder per Callback-URI abgeholt oder als base64 encoded Content innerhalb der SOAP Message übermittelt.

Das Zustellstück wird von MOA-ZS im ersten Schritt auf seine formale Richtigkeit und Vollständigkeit geprüft. Verfügt das Zustellstück über Attachments die mittels Callback Funktion zu holen sind, wird nur der bereits vorhandene Teil des Zustellstücks geprüft. Ist diese Prüfung nicht erfolgreich erhält die Applikation eine entsprechende Fehlermeldung. Im zweiten Schritt wird das Zustellstück persistent gespeichert (Fehlermeldung falls die Speicherung des Zustellstücks nicht möglich ist).

Da es in der Anfangsphase des Dienstes sehr wahrscheinlich ist, dass eine elektronische Zustellung nicht möglich ist, bleibt die Verbindung an dieser Stelle bestehen und MOA-ZS veranlasst eine Prüfung der elektronischen Zustellbarkeit mittels Anfrage an den Zustellkopf [7]. Wurde das Zustellstück mit einer applikationspezifischen bPK übermittelt wird zuvor noch eine bPK Umrechnung beim SZR durchgeführt (siehe Abschnitt 3.1). Kann dem Empfänger elektronisch zuge-

stellt werden, erhält die Applikation an dieser Stelle eine Bestätigung der elektronischen Zustellbarkeit. MOA-ZS beginnt in diesem Fall die Callback Attachments zu laden. Anderenfalls erhält die Applikation eine Nachricht die besagt, dass die elektronische Zustellung mangels elektronischer Adressierbarkeit des Empfängers nicht durchgeführt werden kann. Die Verbindung mit der Applikation wird danach in beiden Fällen getrennt.

Sind die Callback Attachments geladen erhält die Applikation eine Empfangsbestätigung. Können die Attachments nicht geladen werden ergeht eine entsprechende Fehlermeldung (Beides konfigurierbar optional).

Erhält die aufrufende Applikation weder eine positive, noch eine negative Antwort ist davon auszugehen, dass die Antwort negativ ist, aber nicht übermittelt werden konnte.

Scheitert die elektronische Zustellung an einem späteren Punkt infolge eines technischen Gebrechens, wird eine entsprechende Fehlermeldung an die Applikation gerichtet.

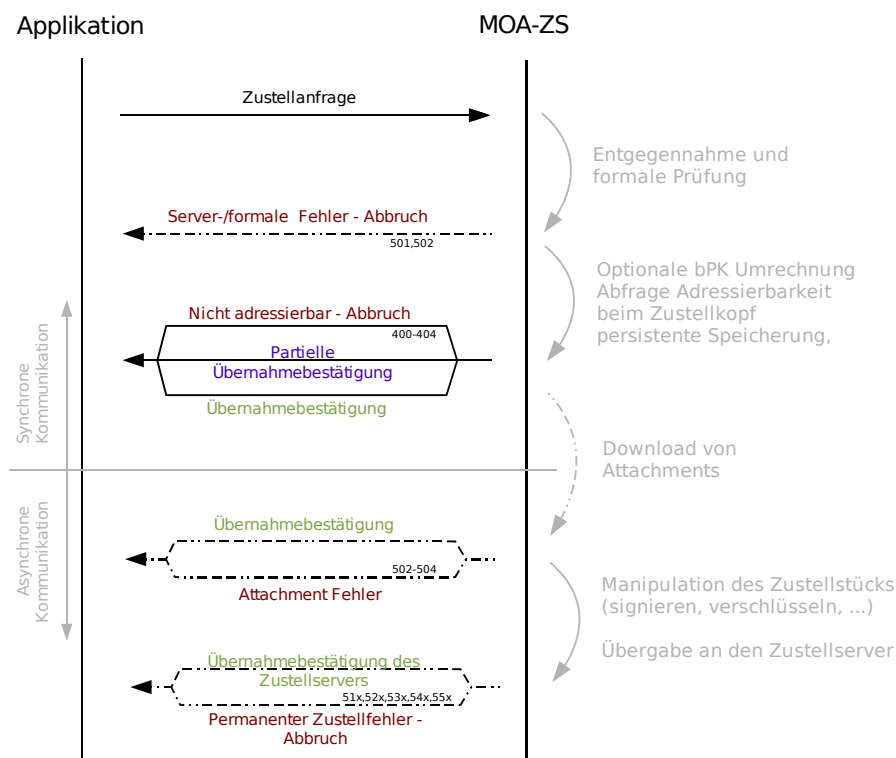


Abbildung 2.1: Kommunikationsablauf zwischen Applikation und MOA-ZS.

Es ist zu beachten, dass die Kommunikation zwischen Applikation und MOA-ZS aus synchronen und asynchronen Komponenten besteht. Im Falle (nur bei Callback-Attachments) der asynchronen Kommunikation, wechseln SOAP Client und SOAP Server die Rollen: Jetzt ist MOA-ZS der Client und die Applikation der Server. Welche SOAP-Nachrichten zu welchem Zeitpunkt ausgetauscht werden ist

Abbildung 2.2 zu entnehmen.

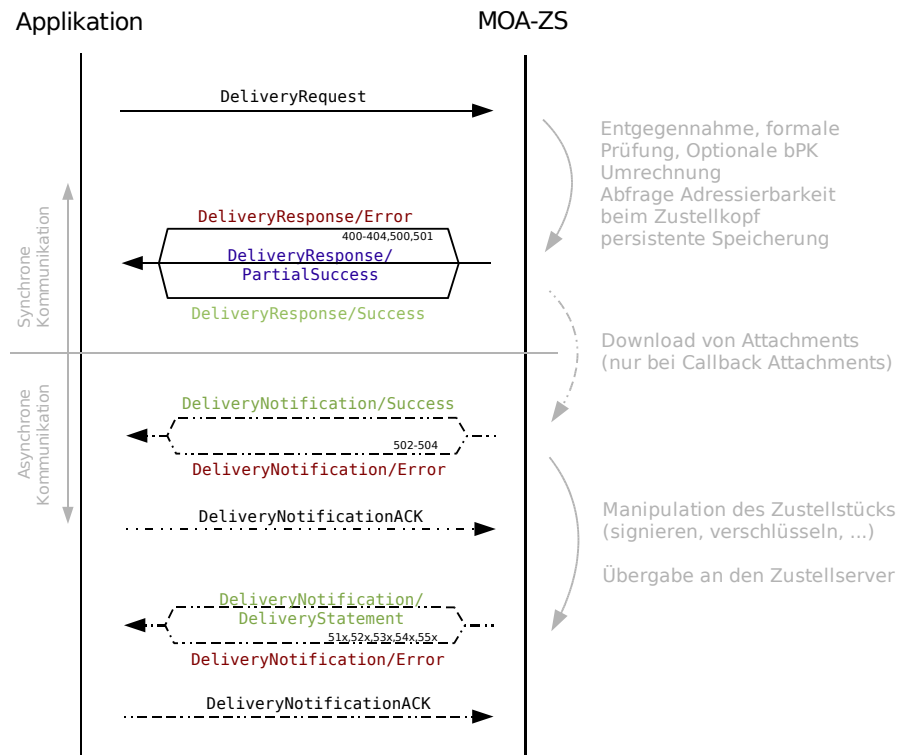


Abbildung 2.2: Kommunikationsablauf zwischen Applikation und MOA-ZS mit SOAP-Nachrichten.

Zustellnachweise und Verständigungen über nicht erfolgreiche elektronische Zustellung werden vom jeweiligen Zustelldienst direkt an die Applikation gerichtet (siehe [12]) und sind nicht Gegenstand dieser Spezifikation. 75

Ebenfalls nicht Gegenstand dieser Spezifikation ist die Authentifizierung und Sicherung des Datenverkehrs zwischen Applikation und MOA-ZS. Authentifizierung und Sicherung liegen im Ermessen des Betreibers und können mittels Trusted Network, TLS oder diversen VPN Lösungen realisiert werden. 80

Nach dem vollständigen Einlangen des SOAP Requestes sind folgenden Schritte durchzuführen:

- SOAP Nachricht **DeliveryRequest** auf die Vollständigkeit der enthaltenen Informationen prüfen (siehe 2.5) 85
- zu Schritt 2 (Prüfung der Adressierbarkeit, siehe Abschnitt 3) übergehen.

2.2 Datenformat für die Dokumentenanlieferung (DeliveryRequest)

Für den Datenaustausch kommen SOAP [22] Container zum Einsatz. Anfragen werden gemäß des MOA-ZS Anfrageformats **DeliveryRequest** formuliert, Antworten gemäß des MOA-ZS Antwortformats **DeliveryResponse** das im wesentlichen aus einer kombinierten “Übernahme erfolgreich - Empfänger adressierbar” Erfolgsmeldung oder einer Fehlermeldung/Abbruchmeldung besteht.

Die Applikation übermittelt folgende Informationen an MOA-ZS

- Applikation (Sender)
- Empfänger (Receiver)
- Zustell-Metainformation (MetaData)
- Zustellstück (Payload)

Abbildung 2.3 gibt einen Überblick über das XML Format der Datenanlieferung.

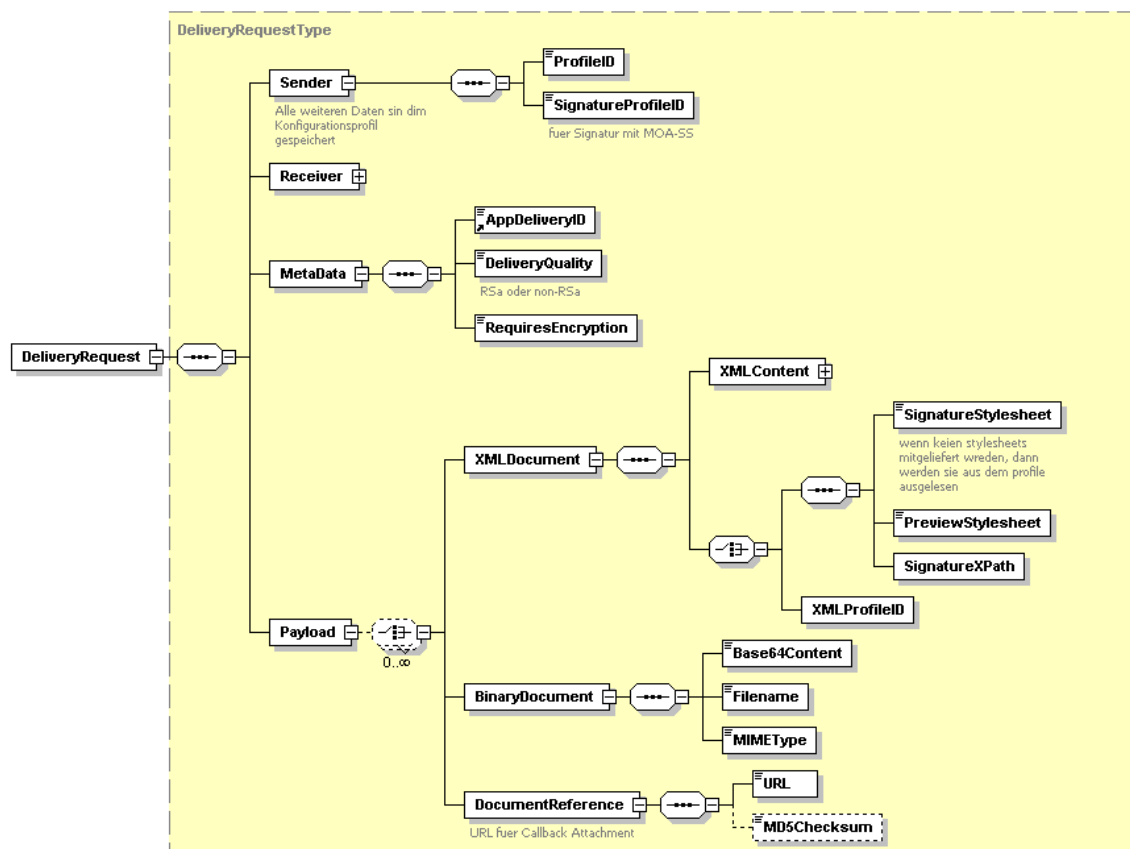


Abbildung 2.3: XML Datenformat für die Datenanlieferung

2.2.1 Applikation (Sender)

Jeder Applikation können ein oder mehrere Profile zugeordnet werden, die über das (`ProfileID`) Element ausgewählt werden. Die Profile beinhalten verschiedene Konfigurationsoptionen, die in Abschnitt 7.1 beschrieben sind. Zusätzlich muss eine Signature-KeyIdentifier (`SignatureKeyID`) für MOA-SS (Signaturserver) angegeben werden (siehe Kapitel 4). 105

2.2.2 Empfänger (Receiver)

Der Empfänger eines Zustellstücks kann auf folgende Arten adressiert werden:

- unverschlüsselte Zustell-bPK des Empfängers
- verschlüsselte Zustell-bPK des Empfängers 110
- bPK des Empfängers in der jeweilig aufrufenden Applikation/Verfahrensbereich
- Name + Verständigungsadresse (+ Geburtsdatum)
- Name + Anschrift (+ Geburtsdatum)
- Stammzahl (= Firmenbuchnummer, Vereinsnummer, ...) für nicht natürliche Personen die mit der unverschlüsselten Zustell bPK gleichzusetzen ist 115

Die Empfängerinformation wird im PersonData2.0 Format [8] übermittelt. Um die Handhabung der PersonData-Datenstruktur zu vereinfachen, kommt eine kompakte Version der PersonData zum Einsatz (`mzs_mypersondata.xsd`, [11]). Abbildung 2.4 zeigt den inneren Aufbau des **Receiver**-Elements.

Tabelle 2.1 illustriert wie bPKs, Firmenbuchnummern, etc. auf das **Identification**-Element gemappt werden. Das **Type**-Element nimmt dabei den Typ der bPK als URN auf, während der eigentliche Wert im zugehörigen **Value**-Element gespeichert wird. Ist keine bPK bekannt, entfällt das **Identification**-Element. 120

bPK	Inhalt des Type-Elements
vZbPK	urn:publicid:gv.at:ecdid+ZS
ZbPK	urn:publicid:gv.at:cdid+ZS
AbPK	urn:publicid:gv.at:cdid+AppVerfahrensbereich
Firmenbuch	urn:publicid:gv.at:baseid+FB
Vereinsregister	urn:publicid:gv.at:baseid+VR
Ergänzungsregister	urn:publicid:gv.at:baseid+ERNNP
nicht nat.Pers.	

Tabelle 2.1: bPKs und ihre Abbildung im Element **Identification/Type**, URN-Werte sind derzeit Richtwerte, genaue Werte zu einem späteren Zeitpunkt laut Verordnung.

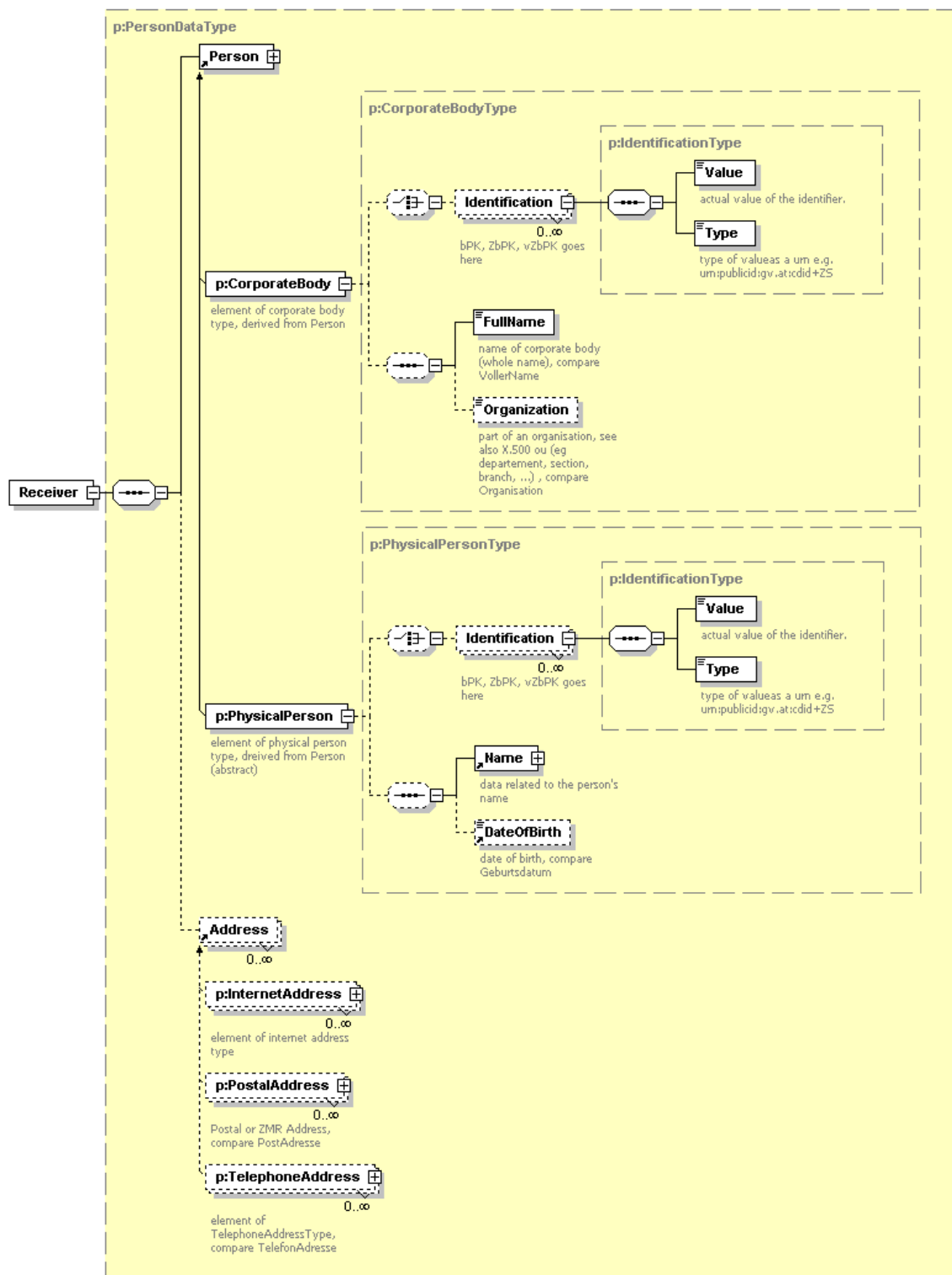


Abbildung 2.4: XML Datenformat für die Datenanlieferung: Angabe des Empfängers

Für natürliche Personen (**PhysicalPerson**) müssen ausserdem im **Name-Element** Vorname (**GivenName**) und Nachname (**FamilyName**) erfasst werden. Für RSa-Zustellungen muss auch noch das Geburtsdatum (**DateOfBirth**) angegeben werden. 125

Für nicht-natürliche Personen (**CorporateBody**) ist das Unterelement **FullName** auszufüllen. Optional kann auch eine Untereinheit der nicht-natürlichen Person angegeben werden (**Organization**). 130

Wenn keine bPK als **Identification-Element** angeführt ist, dann muss mindestens eine Adresse (Post, Internet oder Telefon) angegeben werden. Es können beliebig viele Adressen angegeben werden.

Ein Praxisbeispiel für den Gebrauch von Name, Adresse, etc findet sich in Abschnitt A.1 135

2.2.3 Zustell-Metainformation

Folgende Metainformationen sind für die Zustellung notwendig und müssen angegeben werden:

- Zustell ID (**AppDeliveryID**)
- Zustellungsqualität (RSa, nicht-RSa) (**DeliveryQuality**) 140
- Verschlüsselungspflicht (boolean) (**RequiresEncryption**)

Die *Zustell ID* (**AppDeliveryID**) gibt an unter welcher Kennzahl das Zustellstück der Applikation bekannt ist. *RSa Zustellung* (**DeliveryQuality**) gibt an ob das Zustellstück eine Rsa Zustellung ist. Bei RSa Zustellung muss, sofern der Empfänger über Name und Adresse adressiert wird, das Geburtsdatum zwingend angegeben werden. *Verschlüsselungspflicht* (**RequiresEncryption**) fordert dass eine elektronische Zustellung nur möglich ist wenn der Empfänger einen Public-Key für Verschlüsselung bekannt gegeben hat. 145

2.2.4 Zustellstück – Payload

Ein Zustellstück kann aus einem oder mehreren XML oder binären Dateien bestehen. XML Dateien werden direkt eingebunden **XMLDocument/XMLContent**. 150

Zusätzlich können ein Signatur-Stylesheet (**SignatureStylesheet**) und ein Stylesheet für die Vorschau im Browser (**PreviewStylesheet**) übergeben werden. Wird kein Stylesheet übergeben, so werden Stylesheets die im Profil gespeichert sind benutzt. 155

Binäre Dateien werden entweder im **BinaryDocument** als base64 codierter Inhalt (**Base64Content**) eingebunden, oder als Callback-Attachment unter **DocumentReference** angesprochen. Für base64 codierte Attachments ist zusätzlich der Dateiname (**Filename**) und der MIME-Type (**MIMETYPE**) anzugeben. Wird Callback eingesetzt, so muss der zugehörige HTTP Serverprozess HTTPS und HTTP 1.1 Range Header Fields [3] unterstützen. Für eine externe Datei kann zusätzlich eine MD5 Prüfsumme [20] angegeben werden. 160

2.3 Datenformat für Erfolgs- und Fehlermeldungen in der Dokumentanlieferung (DeliveryResponse)

165

Es gibt zwei Arten von Erfolgsmeldung

- Übernahmebestätigung/adressierbar
- partielle Übernahmenbestätigung (Callback option)

und ein Art von Fehlermeldung, die durch Errorcodes und Errormessages näher spezifiziert wird. Details sind der Abbildung 2.5, den Beispielen 2.3.1, 2.3.2 und 2.3.3, sowie der Tabelle 2.2 zu entnehmen. 170

Fehler 521 “No public key available” tritt nur auf wenn die Applikation bei der Übergabe des Zustellstück die Verschlüsselung gefordert hat (`RequiresEncryption="true"`). Fehler 552 tritt nur auf, wenn der Bürger seinen Account bei einem Zustelldienst löscht, während ein Zustellstück von MOA-ZS bearbeitet wird (ist also sehr unwahrscheinlich). 175

Außer diesen Fehlern, gibt es noch automatisch generierten Fehlermeldungen des HTTP Server (z.B. 500 - internal Server Error) und des SOAP Servers (SOAP Faults über falsches Datenformat etc.). Eine Liste dieser Fehlercodes ist der Dokumentation des MZS beizufügen. 180

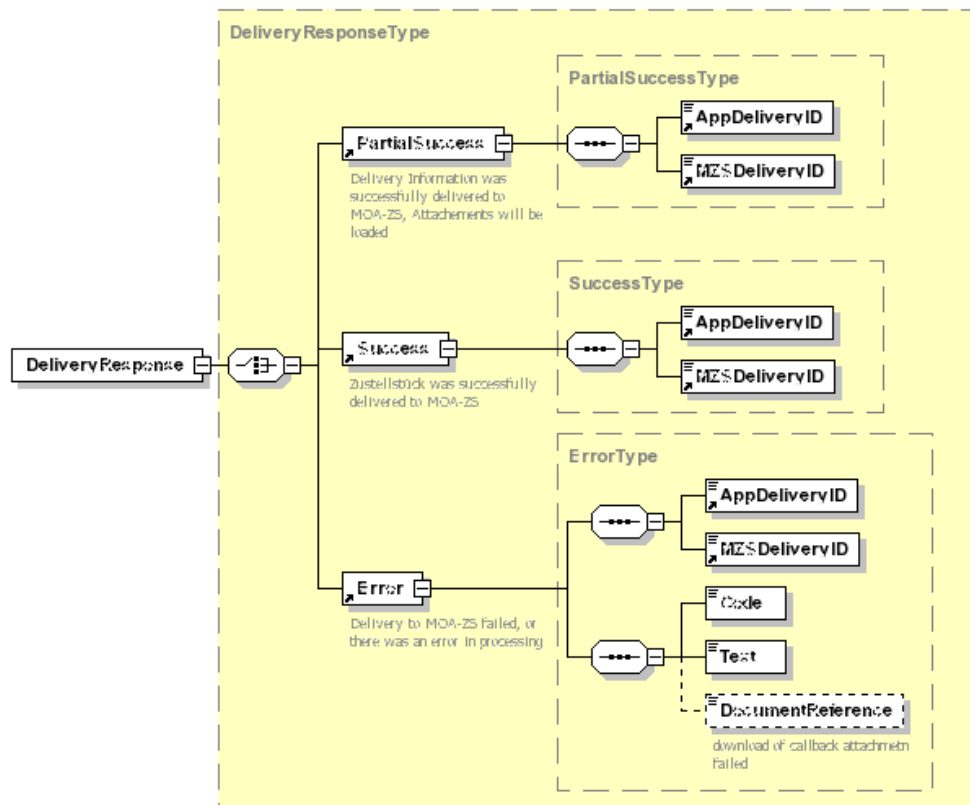


Abbildung 2.5: XML Datenformat für die Antwort auf Datenanlieferung (synchrone Kommunikation): Success, PartialSuccess und Error.

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope"
  xmlns:mzs="http://reference.e-government.gv.at/namespace/moazs10#/"
>
  <SOAP-ENV:Body>
    <mzs:DeliveryResponse>
      <mzs:Success>
        <mzs:AppDeliveryID>1234567</mzs:AppDeliveryID>
        <mzs:MZSDeliveryID>abc765432</mzs:DeliveryID>
      </mzs:Success>
    </mzs:DeliveryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Beispiel 2.3.1: Nachrichtenformat für Erfolgsmeldungen -
Übernahmenbestätigung, Bestätigung der Adressierbarkeit

```

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope"
  xmlns:mzs="http://reference.e-government.gv.at/namespace/moazs10#"
>
  <SOAP-ENV:Body>
    <mzs:DeliveryResponse>
      <mzs:PartialSuccess>
        <mzs:AppDeliveryID>1234567</mzs:DeliveryID>
        <mzs:MZSDeliveryID>abc765432</mzs:DeliveryID>
      </mzs:PartialSuccess>
    </mzs:DeliveryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Beispiel 2.3.2: Nachrichtenformat für Erfolgsmeldungen - partielle
Übernahmenbestätigung im Callback Fall

```

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope"
  xmlns:mzs="http://reference.e-government.gv.at/namespace/moazs10#"
>
  <SOAP-ENV:Body>
    <mzs:DeliveryResponse>
      <mzs:Error>
        <mzs:AppDeliveryID>1234567</mzs:AppDeliveryID>
        <mzs:MZSDeliveryID>abc765432</mzs:DeliveryID>
        <mzs:Code>502</mzs:Code>
        <mzs:Text>Attachment could not be loaded</mzs:Text>
        <mzs:File>foobar.pdf</mzs:File>
      </mzs:Error>
    </mzs:DeliveryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Beispiel 2.3.3: Nachrichtenformat für Fehlermeldungen und Abbruchs-
verständigungen

Code	Error message	File
Fehler in der Adressierbarkeitsprüfung		
400	vZbPK unresolvable	
403	Person unaddressable - too many hits	
404	Person unaddressable - no match found	
Server und formale Fehler – Dokumentanlieferung		
500	Queue congestion	attach. name attach. name
501	Missing metainformation	
502	Attachment could not be loaded	
503	Attachment MD5 verification failed	
504	MOAZS internal server error	
Serverfehler – SZR		
510	SZR is not responding	
511	bPK not valid	
512	bPKDomain not valid	
513	bPK unresolvable	
Serverfehler – Zustellkopf		
520	LDAP head is not responding	
521	No public key available	
Serverfehler – MOA-SS		
530	MOA-SS is not responding	
531	SignatureKeyID unknown	
532	Signature failed	
Serverfehler – Verschlüsselung		
540	Crypto subsystem is not responding	
541	Not a valid X.509 DER public key	
542	Encryption failed	
Serverfehler – Zustelldienst		
550	Deliveryserver is not responding	
551	Data could not be transmittet	
552	Recipient is not known at this server	

Tabelle 2.2: Fehlercodes in der Zustellstückannahme

2.4 Datenformate der Benachrichtigungen (DeliveryNotification)

In der asynchronen Kommunikation (MOA-ZS an Applikation, im Falle von Callback-Attachments) kommen die Nachrichten **DeliveryNotification** (siehe Abbildung 2.6) als Request und **DeliveryNotificationACK** als Response vor. 185

DeliveryNotification ist formatmäßig stark an **DeliveryResponse** angelehnt und die Unterelemente **Success** und **Error** sind analog definiert. Die Fehlermeldungen decken sich ebenfalls mit denen aus **DeliveryResponse** und sind aus Tabelle 2.2 ersichtlich.

DeliveryNotification enthält außerdem das Unterelement **DeliveryStatement**, das übermittelt wird nachdem MOA-ZS das Zustellstück erfolgreich an den Zustelldienst übergeben hat. **DeliveryStatement** beinhaltet den zuständigen Zustelldienst (**DeliveryServer**), den Eingangszeitpunkt (**Timestamp**) und die ID unter der diese Zustellung dort bekannt ist (**ZSDeliveryID**). 195

Die Übermittlung von **DeliveryNotification** ist nicht zwingend erforderlich und kann über das jeweilige MOA-ZS Profil unterdrückt werden. 195

Die Nachricht **DeliveryNotificationACK** dient nur dazu den Erhalt der **DeliveryNotification** Nachrichten zu bestätigen und enthält keine neue Information.

2.5 Vollständigkeitsprüfung

200

Vor der Adressierbarkeitsprüfung muss die Vollständigkeit der Zustell-Angaben geprüft werden. Unvollständige Zustellanträge sind mit dementsprechender Fehlermeldung (Error 501, siehe 2.3) abzuweisen. Die notwendige Zustell-Angaben können Abbildungen 2.3 und 2.4 entnommen werden. Verpflichtende Angaben sind mit durchgehender Linie umrahmt, optionale Angaben mit strichlierter. 205

Ausser dem Name des Empfängers muss entweder ein **Identification-Element** oder ein **Address-Element** angegeben werden.

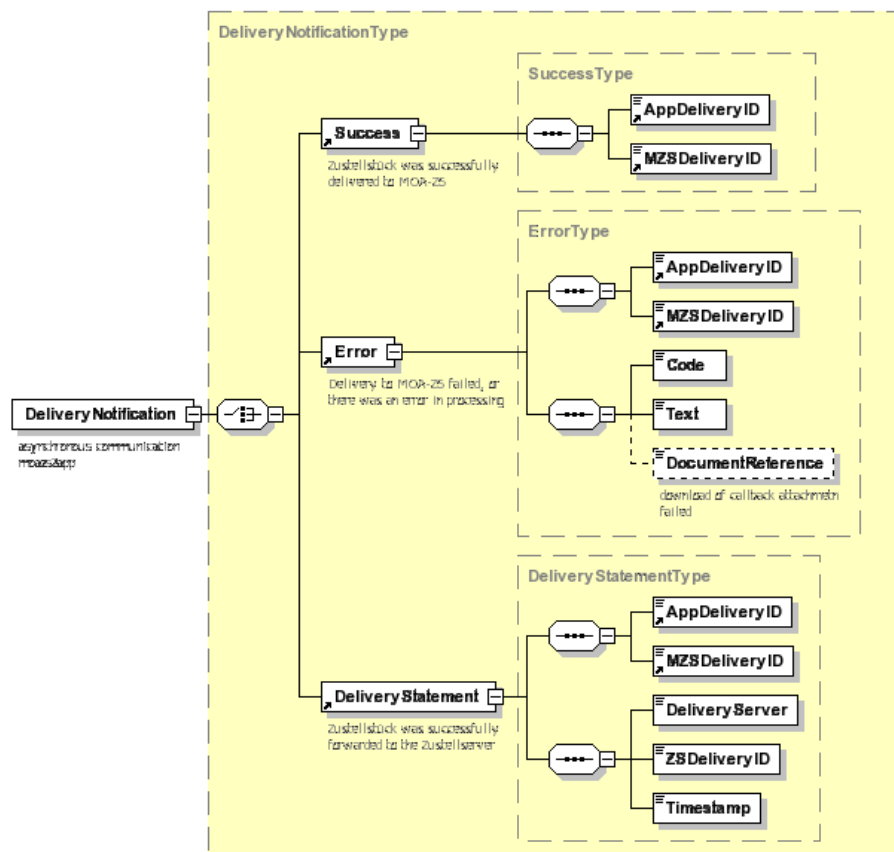


Abbildung 2.6: XML Datenformat für die Verständigungen zwischen MOA-ZS und Applikation (asynchrone Kommunikation): Success, Error und DeliveryStatement.

Kapitel 3

Prüfung der Adressierbarkeit

Die Prüfung der Adressierbarkeit erfolgt in zwei Schritten:

210

1. Falls eine applikationsspezifische bPK (AbPK) als Empfänger angegeben wurde, wird diese vom SZR in eine (verschlüsselte) Zustell-bPK (vZbPK) umgerechnet.
2. Abfrage mittels vZbPK, ZbPK oder Name + Adresse (+ Geburtsdatum) beim Zustellkopf.

215

3.1 bPK Umrechnung beim SZR

Die Kommunikation mit dem SZR erfolgt gemäß der SZR Spezifikation [15]. Da diese noch nicht freigegeben ist, wird der für MOA-ZS relevante Teil im folgenden erläutert. Um eine erfolgreiche Umrechnung durchzuführen, müssen die AbPK Parameter aus der Anfrage und der Public-Key des Zustellkopfes aus der MOA-ZS Konfiguration oder vom Zustellkopf selbst, übergeben werden. Die Schnittstelle zum SRZ ist derzeit noch nicht veröffentlicht. Dieser Abschnitt beschreibt die Schnittstelle soweit sie für MOA-ZS relevant ist.

220

3.1.1 Funktionaler Überblick

Diese Schnittstelle wird nur dann angesprochen, wenn MOA-ZS eine bPK aus einem applikationsspezifischen Bereich übermittelt bekommt. Das schon vorhandene bereichsspezifische Personen Kennzeichen (AbPK) wird vom SZR in ein für die Zustellung geeignetes Personen-Kennzeichen umgewandelt, und mit dem mitzusendenden PublicKey des Zustellkopfes verschlüsselt an MOA-ZS retourniert (Verschlüsselte Zustell-bPK vZbPK). Der PublicKey des Zustellkopfes befindet sich unter der URL <https://zustellung.gv.at/PublicKey>.

225

230

3.1.2 MOA-ZS Anfrage

Die MOA-ZS Anfrage wird in der Spezifikation des Stammzahl-Registers [15] als bPK-Transformation betitelt.

Die für eine eindeutige Umwandlung unbedingt nötigen Daten setzen sich wie folgt zusammen: 235

- Identifikationsdaten

Die Identifikationsdaten, die zu einer eindeutigen Identifikation des Empfängers nötig sind, bestehen aus den folgenden Daten:

- Familienname 240
- Vorname
- Geburtsdatum

- Input-bPK

Das bereichsspezifische Personenkennzeichen, welches von der Fachapplikation an MOA-ZS übergeben wurde. 245

- Input-Bereichskennung

Damit der Stammzahl-Register die Umrechnung in die Zustell-bPK vornehmen kann, muss der Bereich der übergebenen bPK als String mitgeliefert werden.

- Output-Bereich 250

Das Service bPK-Transformation erlaubt die Überführung einer bPK in eine bPK aus einem anderen Bereich. Dazu muss die gewünschte Bereichskennung laut [23] übergeben werden. Es ist erlaubt eine Liste von OutputBereichen und die dazugehörigen PublicKeys zu senden. Für die elektronische Zustellung wird nur die bPK für den Bereich Zustellung benötigt. 255

- Output-Bereichskennung
- PublicKey
Der PublicKey des Zustellkopfes muss mit übergeben werden, um die Verschlüsselung der bPK zu ermöglichen.

Die Element der Anfrage werden durch folgendes Schema beschrieben:

260

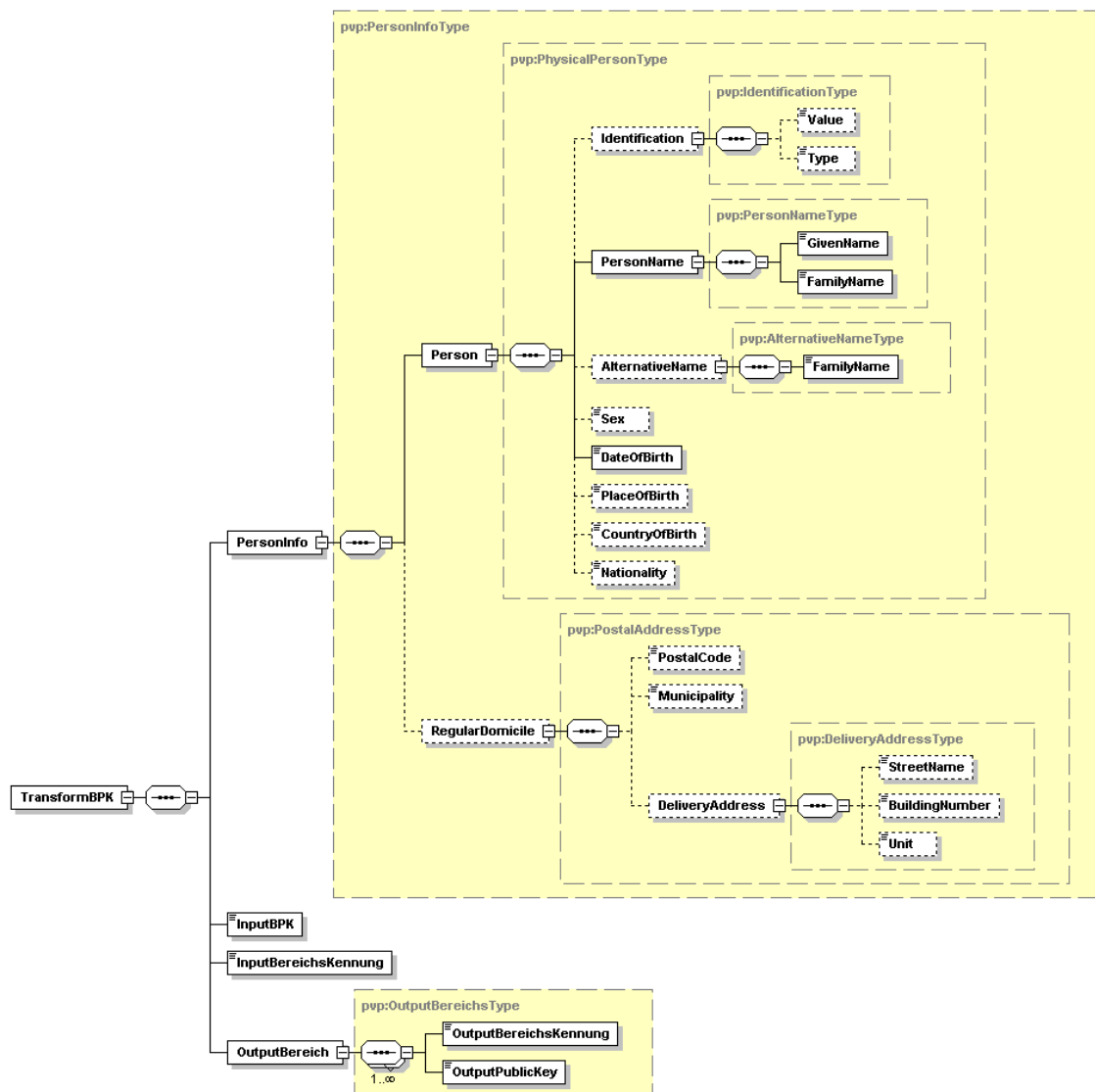


Abbildung 3.1: XML Request-Schema der SZR-Anfrage

3.1.3 Antwort des Stammzahlenregisters

Die Antwort des Stammzahlenregisters ist das Personen-Kennzeichen der Zustellung, welches mit dem PublicKey des ZustellKopfes verschlüsselt ist. Da, wie in obigen Schema ersichtlich, die Outputbereichskennung mehrmals vorkommen kann, besteht die Antwort aus einer Liste von optionalen Type-Value Paaren. Diese Tupel bestehen aus dem String, der den Outputbereich spezifiziert und der für diesen Bereich erstellten und verschlüsselten bPK. Die SOAP Nachricht gehorcht folgendem Schema:

265

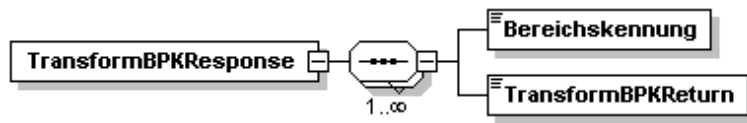


Abbildung 3.2: XML Antwort-Schema der SZR-Anfrage

Die Antwort besteht aus dem TransformBPKReturn-String.

3.2 Adressierbarkeitsanfrage beim Zustellkopf

270

Wurden beim Zustellstück mehrere Empfängeradressen angegeben, so ist für jede Adresse der Zustellkopf gesondert zu befragen (siehe Zustell-Gesetz Gesetz [13]).

Die Kommunikation mit dem Zustellkopf erfolgt gemäß der Interface Spezifikation Applikation ↔ Zustellkopf (ZUSEKOPF, [7]). Die Antwort besteht aus einem Ergebnisset oder einer Fehlermeldung.

275

Wurden mehrere Anfragen gestellt, so sind die Antworten abzuwarten.

Elektronische Postfächer mit Verschlüsselungskey sind zu bevorzugen. Hat die Applikation in der Übergabe des Zustellstücks Verschlüsselungspflicht (`RequiresEncryption="true"`) angegeben so muss eine elektronische Zustellung verneint werden, falls kein elektronisches Postfach mit Verschlüsselungskey gefunden werden kann (Fehler 521, Abschnitt 2.3). Bei mehreren gleichwertigen Postfächern ist eines davon per Zufallszahlenalgorithmus auszuwählen.

280

Ist die elektronische Zustellung möglich, so ist das Zustellstück persistent zu speichern. Erst danach kann eine (partielle) Annahmebestätigung (`DeliveryResponse/Success` oder `PartialSuccess`) (siehe Abschnitt 2.3) an die Applikation geliefert werden.

285

3.3 Callback-Attachments

Wenn das Zustellstück "Callback-Attachments" einsetzt, müssen diese jetzt abgerufen werden. Ist der Download nicht erfolgreich, schlägt die Überprüfung der (optionalen) MD5 Prüfsumme oder die persistente Speicherung fehl, so ist eine entsprechende Fehlermeldung (Fehler 502-504, Abschnitt 2.3) an die Applikation zu senden. War der Download/die Überprüfung erfolgreich so ist nach erfolgter persistenter Speicherung, eine Übernahmebestätigung zu senden.

290

Kapitel 4

Anbringen der Absendersignatur

295

Zustellstücke werden extern via MOA-SS signiert. Die Kommunikation mit MOA-SS erfolgt gemäß der MOA-SS Interface Spezifikation [21]). Soweit die Schnittstelle für die elektronische Zustellung benötigt wird, wurde sie in diesem Dokument beschrieben.

Absendersignaturen sind XMLDsig Signaturen [1] und werden als “enveloped signature” ausgeführt. Im Sinne des SigG Gesetzes [14] wird die Signatur über das Ergebnis aller notwendiger XSLT Transformationen (also die Bürgersicht) gebildet.

4.1 Funktionaler Überblick

Diese Interface Beschreibung ist Teil der MOA-SS Spezifikation und wurde hier speziell für die Bedürfnisse der MOA-ZS Entwicklung ausgelegt und nochmals dokumentiert.

Signaturen werden im XMLDSIG Format [1] als **Enveloped Signature** ausgeführt. Enthält das Zustellstück ein XML Dokument, so wird die Signatur in diesem Dokument angebracht. Bei mehreren XML Dokumenten wird die Signatur im jeweils ersten (Reihenfolge in der Payload bei der Anlieferung) angebracht. Besteht die Zustellung ausschließlich aus binären Dokumenten, so muss zuerst ein XML Deckblatt erstellt werden, in das die Signatur danach eingebettet wird.

Grundsätzlich kann zwischen zwei Möglichkeiten unterschieden werden, um Anhänge durch die Signatur abzudecken

- direkt – bei Fehlen eines Anhangs kann die Signatur nicht mehr geprüft werden
- indirekt (als XMLDSIG **Manifest**) – bei Fehlen eines Anhangs kann die Signatur geprüft werden, die Signatur über den fehlenden Anhang wird dabei ausgeklammert

Innerhalb der elektronischen Zustellung kommt nur die indirekte Signatur von Anhängen zum Einsatz.

4.2 Datenaufbereitung

Bei der Datenaufbereitung sind folgende Fälle zu unterscheiden:

- XML Zustellstück mit Stylesheet(s)
- Nicht XML Zustellstücke (z.B. PDF Dateien)
- Kombination aus den vorhergehenden Optionen

325

Zusätzlich ist der **SignatureKeyID** aus dem Zustellantrag zu ermitteln. Diese wird als **KeyIdentifier** an MOA-SS übermittelt.

4.2.1 XML Nachrichten

330

Jedes XML Zustellstück verfügt prinzipiell über 2 XSLT Stylesheets:

- Signatur Stylesheet für Ausgabe im Secure-Viewer (wird mitsigniert)
- Stylesheet für die Vorschau im Browser (wird nicht signiert)

Die Stylesheets werden entweder bei der Anlieferung mit übergeben, oder durch die **XMLProfileID** referenziert.

335

Das Signatur-Stylesheet wird als **ds:Transform** Inhalt an MOA-SS übergeben. Das Vorschau-Stylesheet wird nicht zur Signatur an MOA-SS übergeben. Kommt ein XML Zustellstück mit nur einem XSLT Stylesheet, so ist dieses ein Signatur-Stylesheet und als Transformation einzubinden.

4.2.2 Nicht XML Zustellstücke

340

Besteht das Zustellstück ausschließlich aus nicht-XML Dateien, so muss zusätzlich ein rudimentäres XML “Zustelldeckblatt” (**DeliveryInformation**) erstellt werden, in das die Signatur eingebracht wird. Das Deckblatt sollte Sender (Detaildaten mittels **ProfileID** ermittelbar), Empfänger und Attachments auflisten. Das Deckblatt folgt dem **DeliveryInformation** Format (siehe Abbildung 4.1, [6]).

345

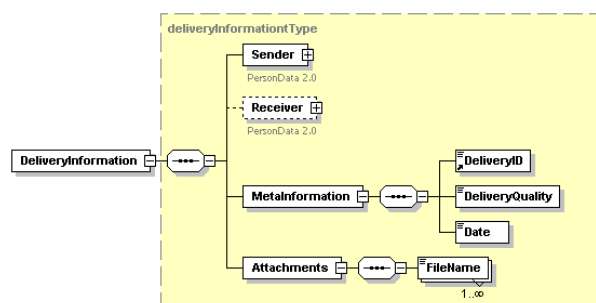


Abbildung 4.1: XML Format für das Attachment Deckblatt


```

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/encoding/">
  <SOAP-ENV:Body>
    <moa:CreateXMLSignatureRequest
      xmlns:moa="http://reference.e-government.gv.at/namespace/moa/20020822#">
      ...
    </moa:CreateXMLSignatureRequest>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Beispiel 4.3.1: SOAP Format der Signaturanfrage

4.2.3 XML und Nicht XML Dateien in Kombination

Müssen mehrere Zustellstückteile signiert werden, so wird die Signatur in den ersten XML Zustellstückteil eingebracht. Alle weiteren Zustellstückteile werden als Manifest signiert.

4.3 MOA-SS Interface: Request

350

MOA-SS wird über seine SOAP[22] Schnittstelle [10] angesprochen (siehe auch Beispiel 4.3.1). Dabei kommen in diesem Fall ausschließlich die Nachrichten `CreateXMLSignatureRequest` (Anfrage) und `CreateXMLSignatureResponse` (Antwort) gemäß des MOA-SS Interface Schemas [10] zum Einsatz.

4.3.1 CreateXMLSignaturRequest

355

Abbildung 4.2 zeigt eine Übersicht über das Request Nachrichtenformat.

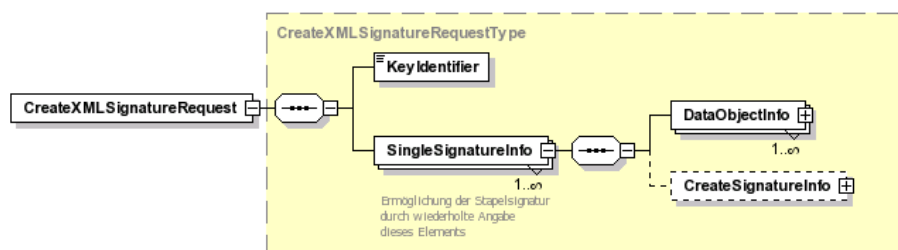


Abbildung 4.2: XML Format für MOA-SS Request (high-level view)

Das Element `CreateXMLSignatureRequest` umschließt die gesamte Anfrage. `KeyIdentifier` gibt den zu verwendenden Signatur-Schlüssel an und liegt bereits als `KeyIdentifier` in der ursprünglichen MOA-ZS Anfrage vor [9].

Das Element `SingleSignatureInfo` umschließt wiederum alle zu signierenden Teile des Zustellstücks und einen Block mit Signatur Metainformation. Im Falle

360

der Zustellung kommt also nur genau ein `SingleSignatureInfo` Element in der SOAP Nachricht vor.

(`CreateSignatureInfo`, Abschnitt 4.3.2).

Die Teile des Zustellstücks werden in `DataObjectInfo` Elemente gekapselt. Die genaue Vorgangsweise ist abhängig vom Typ des Zustellstückteils und wird in Abschnitt 4.3.3 genauer behandelt.

4.3.2 CreateSignatureInfo

Der XML Teil des Zustellstücks, der die Signatur aufnehmen soll (z.B. XML-Bescheid, Deckblatt) wird als Inhalt des Elements `XMLContent` unter `//SingleSignatureInfo/CreateSignatureInfo/CreateSignatureEnvironmentProfile` abgelegt (siehe Abbildung 4.3).

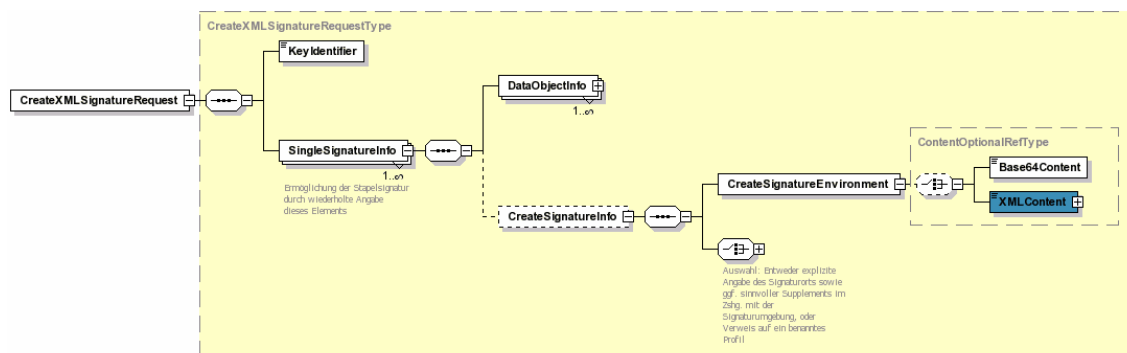


Abbildung 4.3: Das XML Dokument welches die Signatur aufnehmen soll, wird innerhalb von `XMLContent` eingefügt.

Die genaue Position der Signatur wird über `CreateSignatureLocation` unter `//SingleSignatureInfo/CreateSignatureInfo/CreateSignatureEnvironmentProfile` eingefügt (siehe Abbildung 4.4). `CreateSignatureLocation` gibt dabei das Parentelement an. Das `Index` Attribut in `CreateSignatureLocation` an, an welcher Stelle innerhalb des Parentelements die Signatur eingefügt werden soll. Dabei bedeutet `Index="n"`

- `n=0`: als erstes Childelement
- `n>=1`: nach dem n-ten Childelement

```

<foo>
  <bar>
    <!-- wenn Signatur hier, dann
      location = /foo/bar
      index = 0
    -->
    <baz />
    <!-- wenn Signatur hier, dann
      location = /foo/bar
      index = 1
    -->
  </bar>
</foo>

```

Beispiel 4.3.2: Erläuterung von CreateSignaturLocation und seines Index Attributs

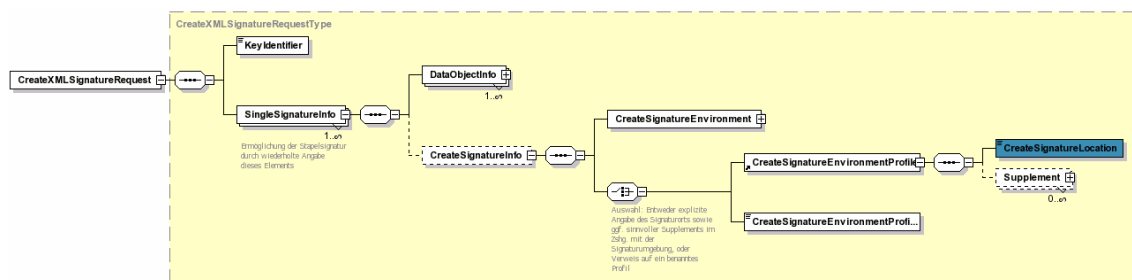


Abbildung 4.4: Die Position, an der die Signatur in das bestehende XML-Dokument eingefügt werden soll, wird über das Element CreateSignaturLocation angegeben. Feinpositionierung erfolgt über das Attribut Index.

4.3.3 DataObjectInfo

Das Vorgehen zum Signieren weiter Zustellstückteile unterscheidet sich je nach Art des Teilstücks:

- XML Dokument das als Envelope für die Signatur dient
- Weitere XML Dokumente
- Nicht-XML Dokumente

385

Allen drei Varianten ist gemein, dass sie unter `//SingleSignatureInfo/DataObjectInfo/DataObject` entweder als content oder als Referenz (über das **Reference** Attribut des **DataObject** Elements) eingebunden werden (siehe Abbildung). Für jedes Dokument wird eine eigenes **DataObjectInfo** Element angelegt.

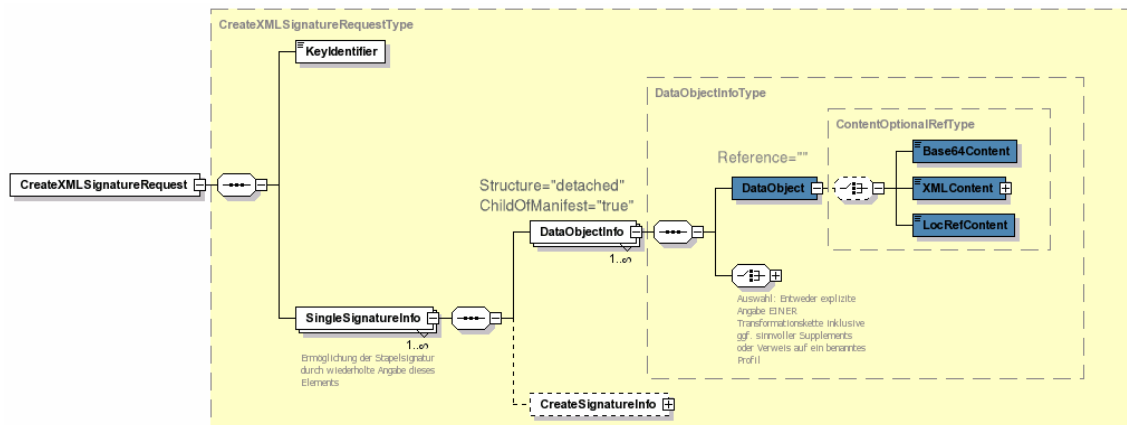


Abbildung 4.5: Zustellstückteile können als Base64content, als XMLContent, als Reference in DataObject oder als LocRefContent eingebunden werden.

XML Envelope

Das Enveloping XML Dokument wird im DataObject Element mit dem Reference Attribut folgendermaßen referenziert:

Reference=""

395

Da eine Signatur über den Envelope auch die Signatur selbst mit einschließen würde, muss der Signaturblock von der Signatur ausgenommen werden. Dies geschieht über eine entsprechende Transformation ("Enveloped Signature Transformation"). Abbildung 4.6 zeigt die Eingliederung von Transformationen. Beispiel 4.3.3 gibt an, wie die Transformation die den Signaturblock von der Signatur ausnimmt, auszusehen hat. Die angewandte Transformation ist ein Standardtransformation aus der XMLDSIG Spezifikation [1]

400

Ein ebenfalls zu signierendes Stylesheet für die Anzeige im Secure-Viewer, wird als Transformation eingebunden.

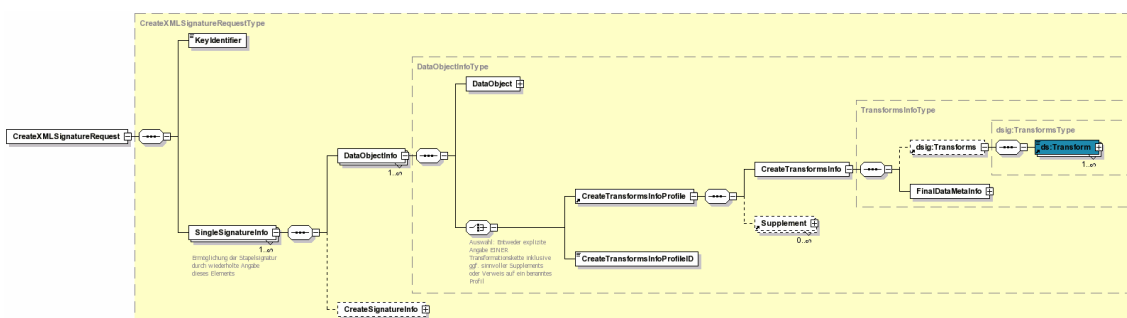


Abbildung 4.6: Es können mehrere Transformationen angegeben werden. Alle Transformationen werden der angegebenen Reihenfolge nach ausgeführt und das endgültige Ergebnis wird signiert.

```

<CreateTransformsInfo>
<dsig:Transforms>
  <dsig:Transform
    Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
  </dsig:Transform>
  <dsig:Transform
    Algorithm="http://www.w3.org/TR/1999/REC-xslt-19991116">
    <xsl:stylesheet xmlns="http://www.w3.org/1999/XSL/Transform">
      <!-- hier xslt templates einfügen -->
    </xsl:stylesheet>
  </dsig:Transform>
</dsig:Transforms>
<FinalDataMetaInfo>
  <MimeType>text/html</MimeType>
</FinalDataMetaInfo>
</CreateTransformsInfo>

```

Beispiel 4.3.3: Die erste Transformation nimmt den Signaturblock von der Signatur aus, die zweite beinhaltet das Stylesheet für die Anzeige in der Bürgerkartenumgebung.

XML Dokumente

405

Unter `//SingleSignatureInfo/DataObjectInfo/DataObject` werden XML Dokumente als Inhalt von `XMLContent` Elementen direkt eingebunden. Soll das XML Dokument in die Signatur eingebettet werden ist das `Structure` Attribut des `DataObjectInfo` Elements auf `enveloping` zu setzen, andernfalls auf `detached`. Das XSLT Stylesheet für die Anzeige in der Bürgerkartenumgebung wird analog zu Abschnitt 4.3.3 im `Transform` Element eingebettet.

410

Nicht-XML Dokumente

In Abhängigkeit von der Größe des Zustellstücks, werden nicht-XML Dokumente entweder unter `//SingleSignatureInfo/DataObjectInfo/DataObject` als Inhalt des `Base64Content` Elements eingebunden oder als `LocRefContent` Element verlinkt. Das `Structure` Attribut des `DataObjectInfo` Elements auf `enveloping` zu setzen. Der Wert des Attributs `ChildOfManifest` des `DataObjectInfo` Elements muss auf `true` gesetzt werden.

415

Die `LocRefContent` Methode setzt voraus, dass MOA-ZS die betreffenden Dateien auf einem HTTP Server ablegen kann. Diese Methode ist für große Zustellstücke (ca. 10 MB) gedacht.

420

In beiden Fällen wird der Wert des `Reference` Attributs des `DataObject` Elements auf den Dateiname des Zustellstückteils (= lokale Filesystem URI) gesetzt. Das ermöglicht es die Signatur auf dem Computer des Empfängers zu prüfen, vorausgesetzt, dass sich Signatur und Anhänge im gleichen Verzeichnis befinden.

425

```

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/encoding/">
  <SOAP-ENV:Body>>
    <moa:CreateXMLSignatureResponse
      xmlns:moa="http://reference.e-government.gv.at/namespace/moa/20020822#">
      ...
    </moa:CreateXMLSignatureResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Beispiel 4.4.1: SOAP Format einer Signaturantwort.

4.4 MOA-SS Interface: Response

Das Antwortformat ist recht einfach gehalten: Das signierte XML Dokument befindet sich im `SignatureEnvironment` Element, Fehler im `ErrorResponse` (siehe Abbildung 4.7). Dabei ist zu beachten, dass nur jene Dokumente inkludiert sind, die beim Request mit `DataObjectInfo@Structure="enveloping"` angegeben wurden. Siehe auch Beispiele 4.4.1 und 4.4.2. 430

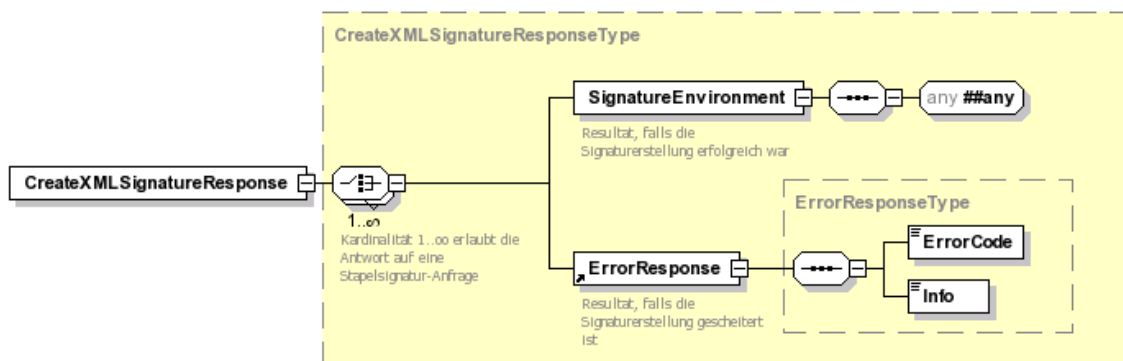


Abbildung 4.7: Im Erfolgsfall befindet sich das signierte XML Dokument im `SignatureEnvironment` Element, Fehler werden als `ErrorResponse` übermittelt.

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/encoding/">
  <SOAP-ENV:Fault>
    <faultcode>SOAP-ENV:Server</faultcode>
    <faultstring>Server Error</faultstring>
    <detail>
      <moa:ErrorResponse
        xmlns:moa="http://reference.e-government.gv.at/namespace/moa/20020822#">
        <ErrorCode>2000</ErrorCode>
        <Info>Fehler im MOA Modul</Info>
      </moa:ErrorResponse>
    </detail>
  </SOAP-ENV:Fault>
</SOAP-ENV:Envelope>
```

Beispiel 4.4.2: SOAP Format eines Signaturfehlers

Kapitel 5

Verschlüsselung des Zustellstücks

Hat der Bürger bei einem Zustelldienst einen Public-Key für die Verschlüsselung im X.509 DER Format[2] hinterlegt, ist dieser unbedingt zu benutzen. Aus Interoperabilitätsgründen, sind vorerst alle Teile des Zustellstücks gesammelt zu verschlüsseln. Schlägt die Verschlüsselung aus irgendeinem Grund fehl, ist die elektronische Zustellung abubrechen und die Applikation diesbezüglich zu verständigen (Fehler 540-542, Abschnitt 2.3). 435

5.1 Zustellstückaufbereitung 440

Bevor die eigentliche Verschlüsselung stattfinden kann, muss das Zustellstück entsprechend aufbereitet werden. Es sind zwei Methoden zu implementieren (siehe Abbildung 5.1):

- multipart-MIME
- als .ZUS File im MIME Container 445

Die jeweils zu benutzende Methode ist über das Konfigurationsfile zu ermitteln (siehe Abschnitt 7.3).

Im Fall von multipart-MIME sind die einzelnen Teile des Zustellstücks (z.B: XML Datei inklusive Signatur, weitere XML Dateien, XSLT Stylesheet(s), PDF Dateien, ...) in einem MIME Container zusammenzufassen. 450

Da das Zustellstück seinen Empfänger möglicherweise als E-Mail erreicht, und dann als Attachment vorliegt, sollte noch ein Mailbody ergänzt werden, der den Bürger auf den Umstand aufmerksam macht, dass es sich bei dieser E-Mail um eine Zustellung handelt, die in Form eines E-Mail Attachments vorliegt. Eine solche Nachricht eignet sich auch um Informationen über den Zustellserver, wie z.B. Hotline unterzubringen. Da gängige E-Mail Programme derzeit nicht mit separat verschlüsselten Attachments zurecht kommen, muss das gesamte Zustellstück verschlüsselt werden. Deshalb kann ein solcher Mailbody nur vor der Verschlüsselung eingefügt werden. Der Mailbody ist mit `content disposition: inline` einzubinden. Die Zustelldienste können unter einer festen 455
460

URL (<http://www.zustellerxy.at/mailbody>) eine “plain text” Datei mit allgemeinen Informationen über die Zustellung und den Zustelldienst bereithalten, die von MOA-ZS eingebunden wird. Ein Beispiel für einen Mailbody findet sich in Abschnitt A.2

Beispiel 5.1.1 zeigt den MIME Envelope eines Zustellstücks ohne Zustellcontainer. 465

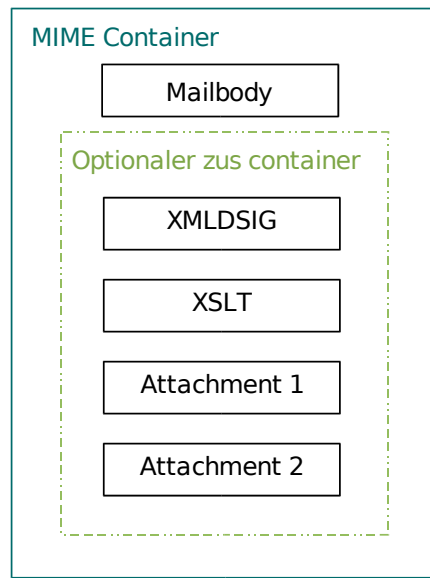


Abbildung 5.1: Zustellstück Container: MIME, ZUS

Das .ZUS File besteht aus einem ZIP-Container [17] mit geänderter Datei Extension (.zus) und eigenem MIME Type (application/vnd.at.zustellung). Dieser Container dient dazu, ein allfälliges auf dem PC des Empfängers installiertes Tool automatisch ansprechen zu können. Dieser Container wird seinerseits in einen MIME Container verpackt. 470

Beispiel 5.1.2 zeigt den MIME-Envelope für den Transfer von .ZUS-Containern.

```
MIME-Version: 1.0
Content-Type: multipart/mixed;
  boundary="-----070603060700010608080604"
Return-Path: test@cio.gv.at
```

```
This is a multi-part message in MIME format.
-----070603060700010608080604
Content-Type: text/plain; charset=us-ascii; format=flowed
Content-Transfer-Encoding: 7bit
```

Das ist eine elektronische Zustellung!

```
-----070603060700010608080604
Content-Type: application/pdf;
  name="test.pdf"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
  filename="test.pdf"
```

```
JVBERi0xLjQNCiXk9tzfdQoxIDAgb2JqDQo8PCAvTGVuZ3RoIDIgMCBsdQogICAvRmlsdGVyIC9G
bGF0ZURlY29kZQ0KPj4NCnN0cmVhbQ0KeJyVVNtqwzAMfQ/4H/RcmGf5GkMZNf37XgjsB7ZujG3Q
...
```

Beispiel 5.1.1: MIME Header für den Transfer von Zustellstückteilen ohne Container

```
MIME-Version: 1.0
Content-Type: multipart/mixed;
  boundary="-----070603060700010608080604"
Return-Path: test@cio.gv.at
```

```
This is a multi-part message in MIME format.
-----070603060700010608080604
Content-Type: text/plain; charset=us-ascii; format=flowed
Content-Transfer-Encoding: 7bit
```

Das ist eine elektronische Zustellung mit .zus Container!

```
-----070603060700010608080604
Content-Type: application/vnd.at.zustellung;
  name="test.zus"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
  filename="test.zus"
a1b2c3d4e5f6g ....
```

Beispiel 5.1.2: MIME Header für den Transfer von .ZUS Containern

5.2 Verschlüsselung

Die Verschlüsselung erfolgt gemäß RSA [5] mit dem Public Key des Empfängers im X.509 DER Format. Die resultierende CMS Datei [4] wird anschließend als S/MIME [18] gekapselt.

Abbildung 5.2 zeigt ein verschlüsseltes Zustellstück und Beispiel 5.2.1 zeigt den MIME-Envelope für den Transfer von verschlüsselten Zustellstücken.

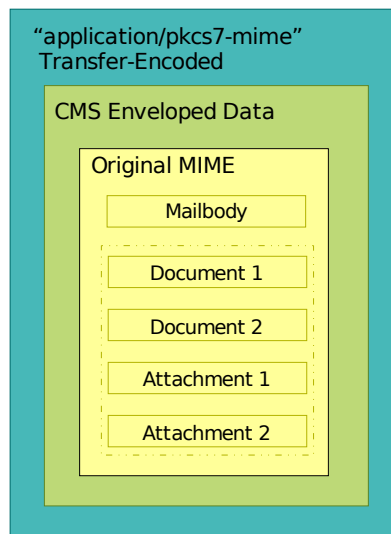


Abbildung 5.2: Zustellstück Container: CMS, S/MIME

MIME-Version: 1.0

Content-Type: application/x-pkcs7-mime; name="smime.p7m"

Content-Transfer-Encoding: base64

```
MIMC7VMGCSqGSIb3DQEHA6CDAu1DMIMC7T4CAQAxggEuMIIBKgIBADCBkjCBizEL
MAkGA1UEBhMCVQVQxSDBGBgNVBAoTP0EtVHJ1c3QgR2VzLiBmLiBTaWNoZXJoZWl0
iBTaWNoZXJoZWl0.....
```

Beispiel 5.2.1: MIME Header für den Transfer von CMS Containern

Kapitel 6

Übergabe an einen Zustelldienst

480

Die Übergabe an den Zustelldienst erfolgt gemäß der Interface Spezifikation Applikation \leftrightarrow ZUSE (zusemsg, [12]).

War die Übergabe erfolgreich so ist eine `DeliveryNotification/DeliveryStatement` Nachricht gemäß app2moa Interface (siehe Abschnitt 2.3) an die Applikation zu senden. Im Fehlerfall kommt eine `DeliveryNotification/Error` Nachricht (Error 550 - 552) zum Einsatz.

485

Weitere Kommunikation des Zustelldiensts wird direkt an den Absender übermittelt.

Kapitel 7

Konfiguration

Die technische Implementierung der Konfiguration steht dem Auftragnehmer nach Rücksprache mit dem Auftraggeber frei. 490

Im folgenden wird davon ausgegangen, dass die einzelnen Konfigurationsprofile über einen “Primärschlüssel” namens **ProfileID** identifiziert werden können. Analog dazu kann jede Zustellung innerhalb von MOA-ZS über die **MZSDeliveryID** eindeutig identifiziert werden. 495

7.1 Applikationen

Um den Overhead bei der elektronischen Zustellung möglichst klein zu halten, werden die Parameter für die einzelnen Applikationen MOA-ZS-seitig gespeichert und über die ProfilID abgerufen.

Folgende Information sind zu erfassen: 500

- ProfileID
- Endpoint für Benachrichtigungen (Webservice, Postfach, e-mail))
- Administrator (Name, e-mail, tel)
- Absenderinformation
 - Name der Behörde/Körperschaft (PersonData) 505
 - OID/VKZ ¹der Behörde/Körperschaft (PersonData)
 - Anschrift der Behörde/Körperschaft (PersonData)
- Verständigungsintervalle für nicht-RSa (optional, Defaultwerte wie RSa werden vom Zustelldienst automatisch ergänzt)
- Maximale Verweildauer in MOA-ZS (wie lange soll MOA-ZS versuchen das Zustellstück zu verarbeiten und an den Zustelldienst zu übergeben) 510

¹OID=Object Identifier, VKZ=Verwaltungskennzeichen, dienen der eindeutigen Identifikation einer Behörde

- Verständigung über die Übergabe an Zustelldienst (`DeliveryNotification` ja/nein)

Zusätzlich zu den oben genannten Konfigurationen, können XML Dokumentklassen und ihre Abhängigkeiten (Stylesheets, Signature-XPath) unter einer speziellen `XMLProfileID` angesprochen werden. Folgende Daten sind zu erfassen: 515

- `XMLProfileID`
- Signature-Stylesheet
- Preview-Stylesheet
- Signature-XPath (absoluter XPath zum `ParentElement` der Signature) 520
- Signature-Index (Position, die die Signatur innerhalb des `ParentElements` einnehmen soll)

7.2 Komponenten

Pro externer aufzurufender Komponente (SZR, Zustellkopf, MOA-SS, Zustelldienste) kann folgendes konfiguriert werden 525

- `KomponentenID`
- Name des Moduls
- Endpoint für Service
- Zahl und Intervalle der Wiederholungen bei Systemausfall

7.3 MOA-ZS Interne Konfiguration

530

- Zustellcontainerformat (MIME oder ZUS)
- “Queue congestion” Parametriesierung nach Größe und/oder absoluter Zustellstückzahl

Kapitel 8

Logging

535

Folgende Informationen müssen pro versuchter elektronischer Zustellung erfaßt werden

- ProfileID (“Primärschlüssel” der Konfigurationsdatenbank)
- AppDeliveryID (“Primärschlüssel” der externen Applikation)
- MZSDeliveryID (interner “Primärschlüssel”)
- Eingangstimestamp

540

Im Falle der erfolgreichen Übergabe an den Zustelldienst

- MZSID
- ZustelldienstID
- Ausgangstimestamp

545

Im Falle eines Abbruchs

- MZSID
- Fehlercode
- Abbruchtimestamp

Zusätzlich muss im Falle des Versagens von Komponenten / externen Modulen folgende Information erfaßt werden:

550

- KomponentenID
- Fehlertimestamp
- Fehlercode
- Fehlerklasse

555

Abbrüche und erfolgreiche Zustellungen sind getrennt zu erfassen.

Anhang A

Beispiele

A.1 Beispiel: Anlieferung eines Zustellstücks

```
<?xml version="1.0" encoding="UTF-8"?>
<DeliveryRequest
  xmlns="http://reference.e-government.gv.at/namespace/moazs10/app2mzs#"
  xmlns:p="
    http://reference.e-government.gv.at/namespace/persondata/20020228#"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://reference.e-government.gv.at/namespace/moazs10/app2mzs#
    app2mzs.xsd"
>
  <Sender>
    <ProfileID>123456</ProfileID>
    <SignatureKeyID>1111</SignatureKeyID>
  </Sender>
  <Receiver>
    <p:PhysicalPerson>
      <p:Name>
        <p:GivenName>Hermann</p:GivenName>
        <p:FamilyName>Maier</p:FamilyName>
      </p:Name>
      <p:DateOfBirth>1968-11-02</p:DateOfBirth>
    </p:PhysicalPerson>
    <p:PostalAddress>
      <p:PostalCode>1010</p:PostalCode>
      <p:Municipality>Wien</p:Municipality>
      <p:DeliveryAddress>
        <p:StreetName>Am Graben</p:StreetName>
        <p:BuildingNumber>1</p:BuildingNumber>
      </p:DeliveryAddress>
    </p:PostalAddress>
  </Receiver>
</DeliveryRequest>
```

```
</p:PostalAddress>
<p:InternetAddress>
  <p:Address>hermann.maier@superg.com</p:Address>
</p:InternetAddress>
</Receiver>
<MetaData>
  <AppDeliveryID>4711</AppDeliveryID>
  <DeliveryQuality>RSa</DeliveryQuality>
  <RequiresEncryption>true</RequiresEncryption>
</MetaData>
<Payload>
  <XMLDocument>
    <XMLContent>
      <!-- hier steht ein Bescheid in XML-Format -->
    </XMLContent>
    <XMLProfileID>0815</XMLProfileID>
  </XMLDocument>
  <BinaryDocument>
    <Base64Content>
      <!-- hier steht ein PDF-File in base64 codierter Form -->
    </Base64Content>
    <Filename>rechtsmittel.pdf</Filename>
    <MimeType>application/x-pdf</MimeType>
  </BinaryDocument>
</Payload>
</DeliveryRequest>
```

A.2 Beispiel: Mailbody

Sehr geehrte Damen und Herren,

bei dieser Mail handelt sich um eine elektronische Zustellung,
die auf Ihren Wunsch an Sie per E-Mail übermittelt wurde.
Alle weiteren Informationen finden Sie in der Anlage (Attachment).

Für weitere Fragen wenden Sie sich bitte an Ihren Zustelldienst.

Vielen Dank

Literaturverzeichnis

- [1] Mark Bartel. XML Signature Syntax and Processing. <http://www.w3.org/TR/xmlsig-core/>, 2002.
- [2] S. Chokhani. RFC 3647 –Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. <http://www.ietf.org/rfc/rfc3647.txt>, 2003.
- [3] R. Fielding and et. al. RFC 2616 – Hypertext Transfer Protocol – HTTP/1.1 . <http://www.ietf.org/rfc/rfc2616.txt>, 1999.
- [4] R. Housley. *RFC 3369 – Cryptographic Message Syntax (CMS)*, 2002.
- [5] B. Kaliski. RFC2437 – PKCS #1: RSA Cryptography Specifications. <http://www.ietf.org/rfc/rfc2437.txt>, 1998.
- [6] Gregor Karliner.
- [7] Michael Liehmann. ZUSE Interface Spezifikation – Applikation an LDAP. <http://www.cio.gv.at/onlineservices/basicmodules/delivery11/zusekopf10.pdf>, 2004.
- [8] Larissa Naber. <http://www.cio.gv.at/xxx>, 2004.
- [9] Larissa Naber. MOA-ZS Interface Spezifikation – Applikation an MOA-ZS. <http://www.cio.gv.at/onlineservices/basicmodules/moa/zs/app2mzs.pdf>, 2004.
- [10] Larissa Naber. MOA-ZS: XSD Schema der Schnittstelle zwischen Applikation und MOA-ZS, 2004.
- [11] Larissa Naber. PersonData 2.0 Blueprint for app2mzs Interface. http://www.cio.gv.at/moa/zs/mzs_mypersondata_en.xsd, 2004.
- [12] Larissa Naber and Michael Liehmann. ZUSE Interface Spezifikation – Applikation an Zustellserver. <http://www.cio.gv.at/onlineservices/delivery11/zusemsg.pdf>, 2004.
- [13] NN. E-GOV Gesetz, 2004.
- [14] nn. Signatur gesetz, 2004.

- [15] nn. Szt specifikation, 2004.
- [16] OASIS. SAML 1.0 specification. <http://www.oasis-open.org/committees/download.php/2290/oasis-%20saml-1.0.zip>, 2002.
- [17] PKWARE Inc. .zip file format specification. http://www.pkware.com/products/enterprise/white_papers/appnote.html, 2004.
- [18] B. Ramsdell. RFC 2633 – s/mime version 3 message specification. <http://www.ietf.org/rfc/rfc2633.txt>, 1999.
- [19] Peter Reichstaedter and Larissa Naber. Einführung in die elektronische Zustellung. <http://www.cio.gv.at/onlineservices/delivery11/intro.pdf>, 2004.
- [20] R. Rivest. *RFC 1321 – The MD5 Message-Digest Algorithm*, 1992.
- [21] Rudolf Schamberger. Spezifikation Module für Online Anwendungen SP und SS. http://www.cio.gv.at/onlineservices/basicmodules/moa/specification/MOA-SPSS-1.1_20030630.pdf, 2003.
- [22] W3C. Simple Object Access Protocol (SOAP) 1.1. <http://www.w3c.org/TR/2000/NOTE-SOAP-20000508/>, 2000.
- [23] Wiesner. Verfahren und Leistungsbereiche. <http://reference.e-government.gv.at/>, 2003.