

| MOA-ZS<br>Technische Spezifikation |  | Konvention                  |
|------------------------------------|--|-----------------------------|
|                                    |  | mzsspec-1.2.2               |
|                                    |  | Entwurf Öffentlich          |
| Kurzbeschreibung                   | Diese Spezifikation beschreibt die Abläufe innerhalb MOA-ZS und die Datenübergabe an MOA-ZS. |                             |
| Autor(en):                         | Arne Tauber  | Projektteam / Arbeitsgruppe |
|                                    |  | AG-ZUSE                     |
| Beiträge von:                      | Michael Liehmann, Larissa Naber, ...   |                             |

|                |                   |   |                   |
|----------------|-------------------|---|-------------------|
| Version ... :  | <b>TT.MM.JJJJ</b> | Fristablauf:                                    | <b>TT.MM.JJJJ</b> |
| Abgelehnt von: |                   | (Länderangabe bei ablehnender<br>Stellungnahme) |                   |

|                     |                   |   |                   |
|---------------------|-------------------|---|-------------------|
| Unter-Version ... : | <b>TT.MM.JJJJ</b> | Fristablauf:                                    | <b>TT.MM.JJJJ</b> |
|                     |                   | (Länderangabe bei ablehnender<br>Stellungnahme) |                   |

|                      |                   |                              |                   |
|----------------------|-------------------|------------------------------|-------------------|
| Detail-Version ... : | <b>TT.MM.JJJJ</b> | Freigabe:                    | <b>TT.MM.JJJJ</b> |
|                      |                   | (Detailangaben zur Freigabe) |                   |

# Inhaltsverzeichnis

|   |  |    |
|---|--|----|
| 1   | Historie .....   | 4  |
| 2   | Einleitung .....   | 5  |
| 3   | Zustellstückannahme .....  | 7  |
| 3.1   | Funktionaler Überblick .....   | 7  |
| 3.2   | Datenformat für die Dokumentenanlieferung (DeliveryRequest) .....                                | 10 |
| 3.2.1   | Semisynchrone Zustellung .....   | 12 |
| 3.2.2   | Zustellserver (optional) .....   | 12 |
| 3.2.3   | Applikation (Sender) .....   | 13 |
| 3.2.4   | Empfänger (Receiver) .....   | 13 |
| 3.2.5   | Zustell-Metainformation .....  | 15 |
| 3.2.6   | Zustellstück – Payload .....   | 16 |
| 3.3   | Datenformat für Erfolgs- und Fehlermeldungen in der Dokumentanlieferung (DeliveryResponse) ..... | 16 |
| 3.4   | Datenformate der Benachrichtigungen (DeliveryNotification) .....                                 | 19 |
| 3.5   | Vollständigkeitsprüfung .....  | 20 |
| 4   | Prüfung der Adressierbarkeit .....   | 22 |
| 4.1   | bPK Umrechnung beim SZR .....  | 22 |
| 4.1.1   | Funktionaler Überblick .....   | 22 |
| 4.1.2   | MOA-ZS Anfrage .....   | 22 |
| 4.1.3   | Antwort des Stammzahlenregisters .....   | 23 |
| 4.1.4   | Adressierbarkeitsanfrage beim Zustellkopf .....  | 23 |
| 4.1.5   | Callback-Attachments .....   | 23 |
| 5   | Anbringen der Absendersignatur .....   | 24 |
| 5.1   | Funktionaler Überblick .....   | 24 |
| 5.2   | Datenaufbereitung .....  | 24 |
| 5.2.1   | XML Nachrichten .....  | 24 |
| 5.2.2   | Binäre Zustellstücke .....   | 25 |
| 5.2.3   | XML und binäre Dateien in Kombination .....  | 25 |
| 5.3   | MOA-SS Schnittstelle: Request .....  | 25 |
| 5.3.1   | CreateXMLSignaturRequest .....   | 26 |
| 5.3.2   | CreateSignatureInfo .....  | 26 |
| 5.3.3   | DataObjectInfo .....   | 27 |
| 5.4   | MOA-SS Interface: Response .....   | 29 |
| 6   | Verschlüsselung des Zustellstücks .....  | 31 |
| 6.1   | Zustellstückaufbereitung .....   | 31 |
| 6.2   | Verschlüsselung .....  | 32 |
| 7   | Übergabe an einen Zustelldienst .....  | 34 |
| 8   | Konfiguration .....  | 35 |
| 8.1   | Applikationen .....  | 35 |
| 8.2   | Komponenten .....  | 35 |
| 8.3   | MOA-ZS Interne Konfiguration .....   | 36 |
| 9   | Logging .....  | 37 |
| Anhang A  | .....  | 38 |
| Beispiele   | .....  | 38 |
| A.1 Beispiel: Anlieferung eines Zustellstücks ..... | 38   |    |
| A.2 Beispiel: Mailbody .....                        | 38   |    |
| Referenzen .....                                    | 39   |    |

## Abbildungsverzeichnis

|   |    |
|---|----|
| Abbildung 2.1 Überblicksdarstellung des MOA-ZS Workflows .....  | 5  |
| Abbildung 3.1 Kommunikationsablauf zwischen Applikation und MOA-ZS. ....  | 8  |
| Abbildung 3.2 Kommunikationsablauf zwischen Applikation und MOA-ZS mit .....  | 9  |
| Abbildung 3.4 XML Datenformat für die Datenanlieferung: Angabe des .....  | 14 |
| Abbildung 3.5 XML Datenformat für die Antwort auf Datenanlieferung (synchrone .....   | 17 |
| Abbildung 3.6 XML Datenformat für die Verständigungen zwischen MOAZS .....  | 20 |
| Abbildung 5.1: XML Format für das Attachment Deckblatt .....  | 25 |
| Abbildung 5.2: XML Format für MOA-SS Request (high-level view).....   | 26 |
| Abbildung 5.3: Das XML Dokument welches die Signatur aufnehmen soll, wird.....  | 26 |
| Abbildung 5.4: Die Position, an der die Signatur in das bestehende XML-Dokument eingefügt werden soll, wird über das Element <code>CreateSignatureLocation</code> angegeben. Feinpositionierung erfolgt über das Attribut <code>Index</code> . .... | 27 |
| Abbildung 5.5: Zustellstückteile können als <code>Base64Content</code> , als <code>XMLContent</code> oder als <code>LocRefContent</code> eingebunden werden.....  | 28 |
| Abbildung 5.6: Es können mehrere Transformationen angegeben werden. Alle Transformationen werden der angegebenen Reihenfolge nach ausgeführt und das endgültige Ergebnis wird signiert. ....  | 28 |
| Abbildung 5.7: Im Erfolgsfall befindet sich das signierte XML Dokument im <code>SignatureEnvironment</code> Element, Fehler werden als <code>ErrorResponse</code> übermittelt. ....   | 30 |
| Abbildung 6.1: Zustellstück Container: MIME, ZUS .....  | 31 |
| Abbildung 6.2: Zustellstück Container: CMS, S/MIME .....  | 33 |

# 1 Historie

| Version | Autor       | Änderungen  |
|---------|-------------|---|
| 1.1.0   | Arne Tauber | <ul style="list-style-type: none"><li>• Request - Server Element (keine Zustellkopfabfrage) – siehe Abschnitt 3.2.1</li><li>• Konfiguration – Angabe einer Liste von bevorzugten Zustellservern</li></ul> |
| 1.2.0   | Arne Tauber | <ul style="list-style-type: none"><li>• Privatzustellung</li><li>• Semisynchrone Zustellung</li><li>• Duale Zustellung</li><li>• Zustellbenachrichtigungen</li></ul>                                      |
| 1.2.2   | Arne Tauber | <ul style="list-style-type: none"><li>• Mailbody Element</li></ul>  |

## 2 Einleitung

MOA-ZS ist eine Middleware, deren Aufgabe darin besteht, Fachapplikationen den Zugang zur elektronischen Zustellung zu erleichtern, indem 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 8) 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 2.1.

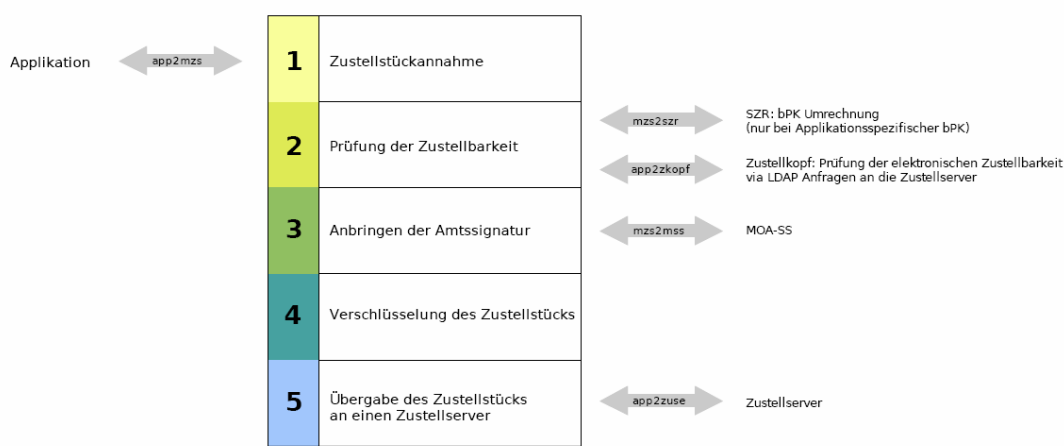


Abbildung 2.1 Überblicksdarstellung des MOA-ZS Workflows

Die vorliegende Spezifikation behandelt nur den Ablauf innerhalb von MOAZS und die Schnittstelle zur Datenübernahme, sowie die Schnittstellen zum Stammzahlregister (SZR) und MOA-SS, soweit sie für die Implementierung von MOA-ZS relevant sind. Der Datenaustausch mit anderen externen Anwendungen wird in den jeweiligen Schnittstellenspezifikationen beschrieben. Das XML-Schema `app2mzs.xsd` ist ein normativer Teil dieser Spezifikation. Einen (nichtnormativen) Überblick über die elektronische Zustellung und das Zustellungseinbindungsmodul MOAZS bietet die Einführung zur elektronischen Zustellung [ZUSEMOD].

In den kommenden Kapiteln findet sich eine Vielzahl von unterschiedlichen Namespaces.

Nachfolgend eine kurze Aufstellung:

| Präfix | Erläuterung    | Namespace   |
|--------|----------------|---|
| dsig   | XMLDSIG        | <a href="http://www.w3.org/2000/09/xmldsig#">http://www.w3.org/2000/09/xmldsig#</a>   |
| soap   | SOAP           | <a href="http://www.w3.org/2001/12/soap">http://www.w3.org/2001/12/soap</a>   |
| saml   | SAML Assertion | <a href="urn:oasis:names:tc:SAML:1.0:assertion">urn:oasis:names:tc:SAML:1.0:assertion</a>   |
| p      | PersonData 2.0 | <a href="http://reference.e-government.gv.at/namespace/persondata/en/20020228">http://reference.e-government.gv.at/namespace/persondata/en/20020228</a> |
| zs     | ZUSE Message   | <a href="http://reference.egovernment.gv.at/namespace/zuse11#">http://reference.egovernment.gv.at/namespace/zuse11#</a>                                 |

| Präfix | Erläuterung             | Namespace   |
|--------|-------------------------|---|
| mzs    | MOA-ZS<br>Schnittstelle | <a href="http://reference.egovernment.gv.at/namespace/moazs10/app2mzs#">http://reference.egovernment.gv.at/namespace/moazs10/app2mzs#</a> |

## 3 Zustellstückannahme

### 3.1 Funktionaler Überblick

Die Übernahme eines Zustellstücks erfolgt gemäß der Applikation ↔ MOA-ZS Schnittstelle zur Datenanlieferung (siehe Abschnitt 3.2). Alle verwendeten XML Nachrichten sind gemäß dem XML Schema app2mzs.xsd 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 8.3).

MOA-ZS übernimmt das Zustellstück von der aufrufenden Applikation. Das dabei zu verwendende Datenformat ist MOA-ZS Message DeliveryRequest (siehe Abschnitt 3.2). Bei dem Zustellstück handelt es sich um eine SOAP Message [SOAP]. 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 referenziert oder als Base64-kodierter Inhalt 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 [ZUSEKOPF]. Wurde das Zustellstück mit einem applikationsspezifischen bPK übermittelt, wird zuvor noch eine bPK-Umrechnung beim SZR durchgeführt (siehe Abschnitt 4.1). Ist der Empfänger elektronisch adressierbar, erhält die Applikation an dieser Stelle eine Bestätigung der elektronischen Zustellbarkeit. MOA-ZS beginnt in diesem Fall vorhandene Callback Attachments zu laden. Anderenfalls erhält die Applikation eine Fehlermeldung, dass die elektronische Zustellung mangels elektronischer Adressierbarkeit des Empfängers nicht durchgeführt werden kann. Die Verbindung mit der Applikation wird anschließend 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 retourniert.

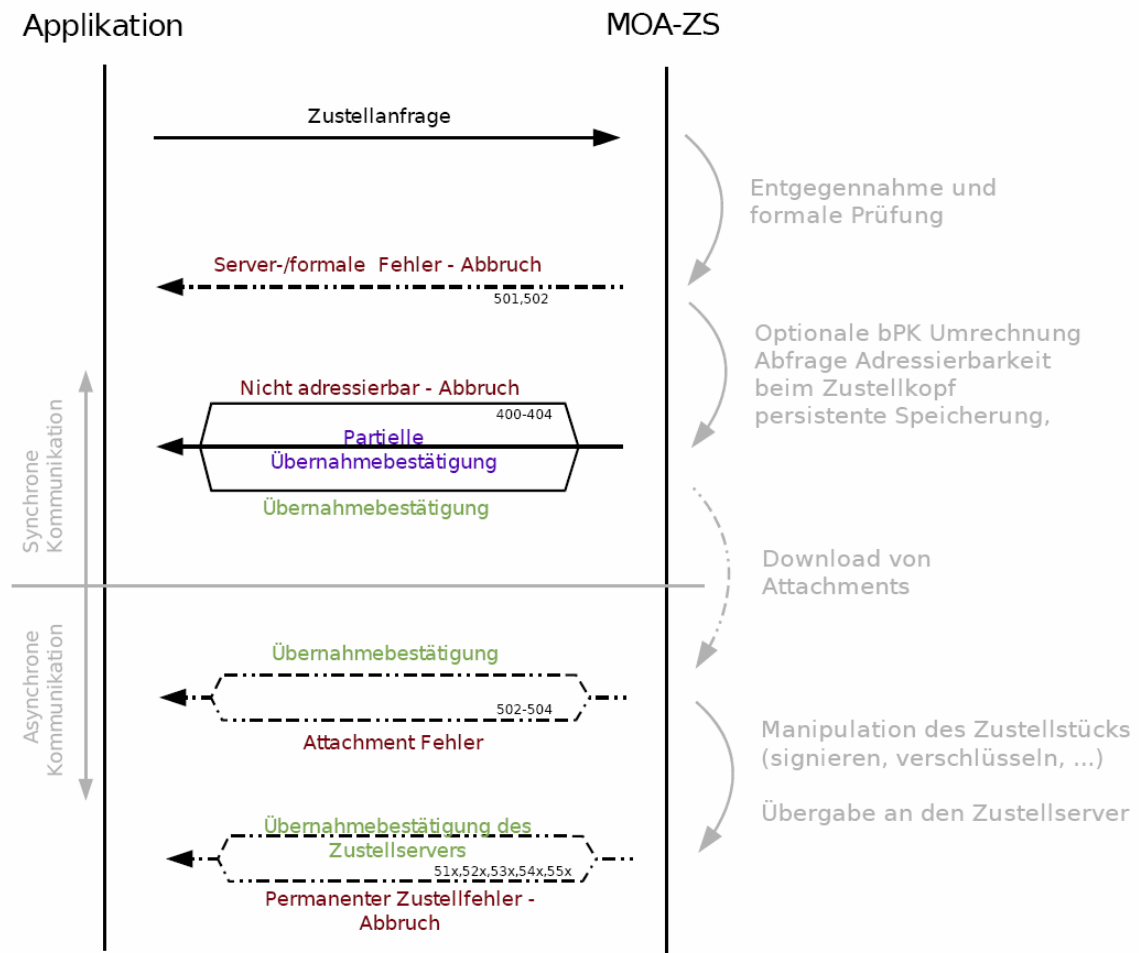


Abbildung 3.1 Kommunikationsablauf zwischen Applikation und MOA-ZS.

Es ist zu beachten, dass die Kommunikation zwischen Applikation und MOA-ZS aus synchronen und asynchronen Komponenten besteht. Welche SOAP-Nachrichten zu welchem Zeitpunkt ausgetauscht werden ist Abbildung 3.1 zu entnehmen.

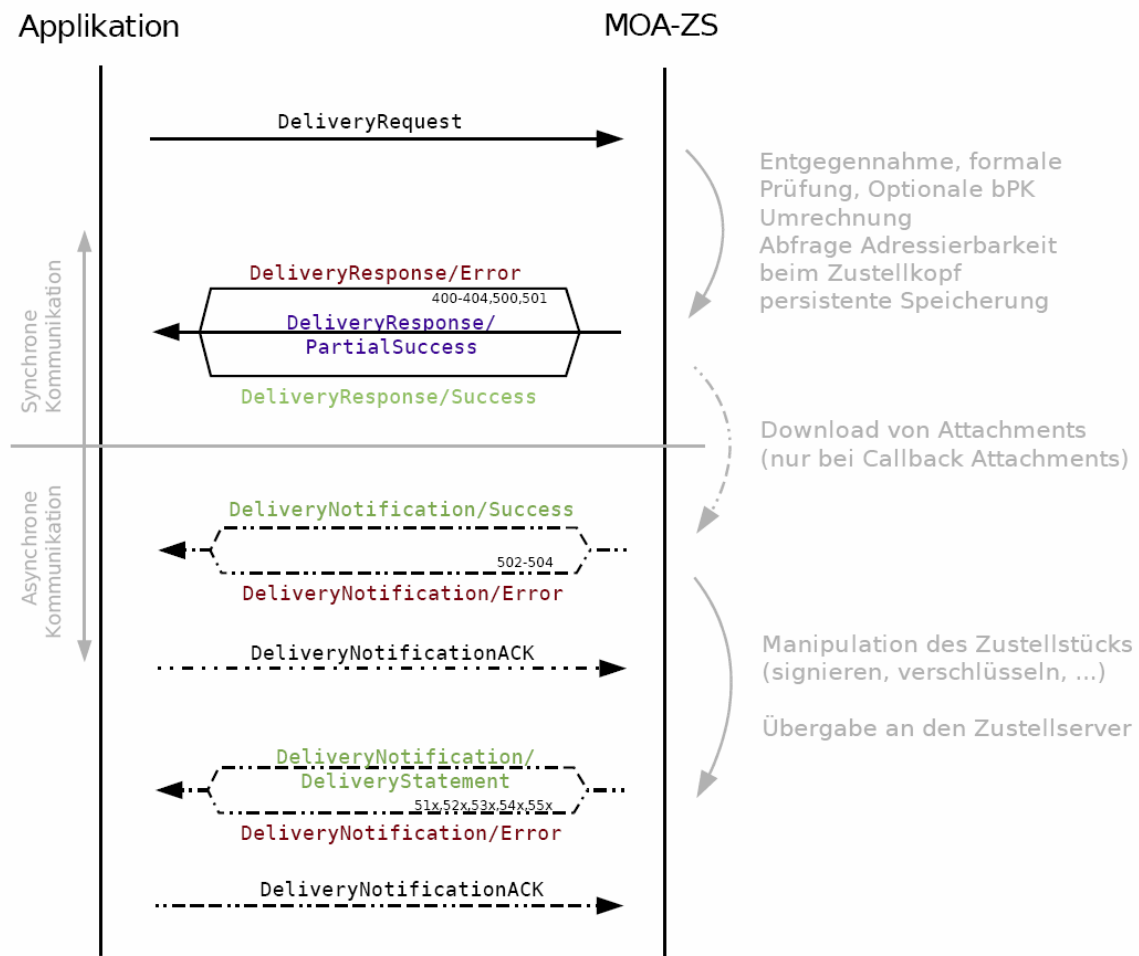


Abbildung 3.2 Kommunikationsablauf zwischen Applikation und MOA-ZS mit SOAP-Nachrichten.

Zustellnachweise und Verständigungen über eine nicht erfolgreiche elektronische Zustellung werden vom jeweiligen Zustelldienst direkt an die Applikation oder an MOA-ZS retourniert, falls MOA-ZS entsprechend konfiguriert wurde (siehe [ZUSEMSG]).

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.

Nach dem vollständigen Einlangen des SOAP Requests sind folgende Schritte durchzuführen:

- SOAP Nachricht auf XML Schemakonformität prüfen (optional konfigurierbar)
- SOAP Nachricht DeliveryRequest auf die Vollständigkeit der enthaltenen Informationen prüfen (siehe Abschnitt 3.5)
- zu Schritt 2 (Prüfung der Adressierbarkeit, siehe Abschnitt 4) übergehen.

Weiters muss MOA-ZS eine optionale (ob aktiv in der Konfiguration definierbar) Schnittstelle zum Empfangen von Zustellbenachrichtigungen implementieren. Diese Schnittstelle muss sowohl Benachrichtigungen von MOA-ZS als auch Übernahmenbestätigungen (RSa Rückscheine) seitens der Zustellserver verarbeiten können.

Nach dem vollständigen Einlangen des SOAP Requests der Zustellbenachrichtigung sind folgende Schritte durchzuführen:

- SOAP Nachricht auf XML Schemakonformität prüfen (optional konfigurierbar)
- Im Falle einer Übernahmebestätigung (RSa Rückschein) muss die Signatur des Zustellserver validiert werden. Die Validierung kann bspw. mit MOA-SP erfolgen.
- Über eine entsprechende Schnittstelle sollen modular Plugins für die Weiterverarbeitung der Zustellbenachrichtigungen eingebunden werden können, welche die weitere Verarbeitung der Zustellbenachrichtigung erledigen.

### 3.2 Datenformat für die Dokumentenanlieferung (DeliveryRequest)

Für den Datenaustausch kommen SOAP Container zum Einsatz (siehe [SOAP]). 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:

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

Abbildung 3.3 gibt einen Überblick über das XML Format der Datenanlieferung.

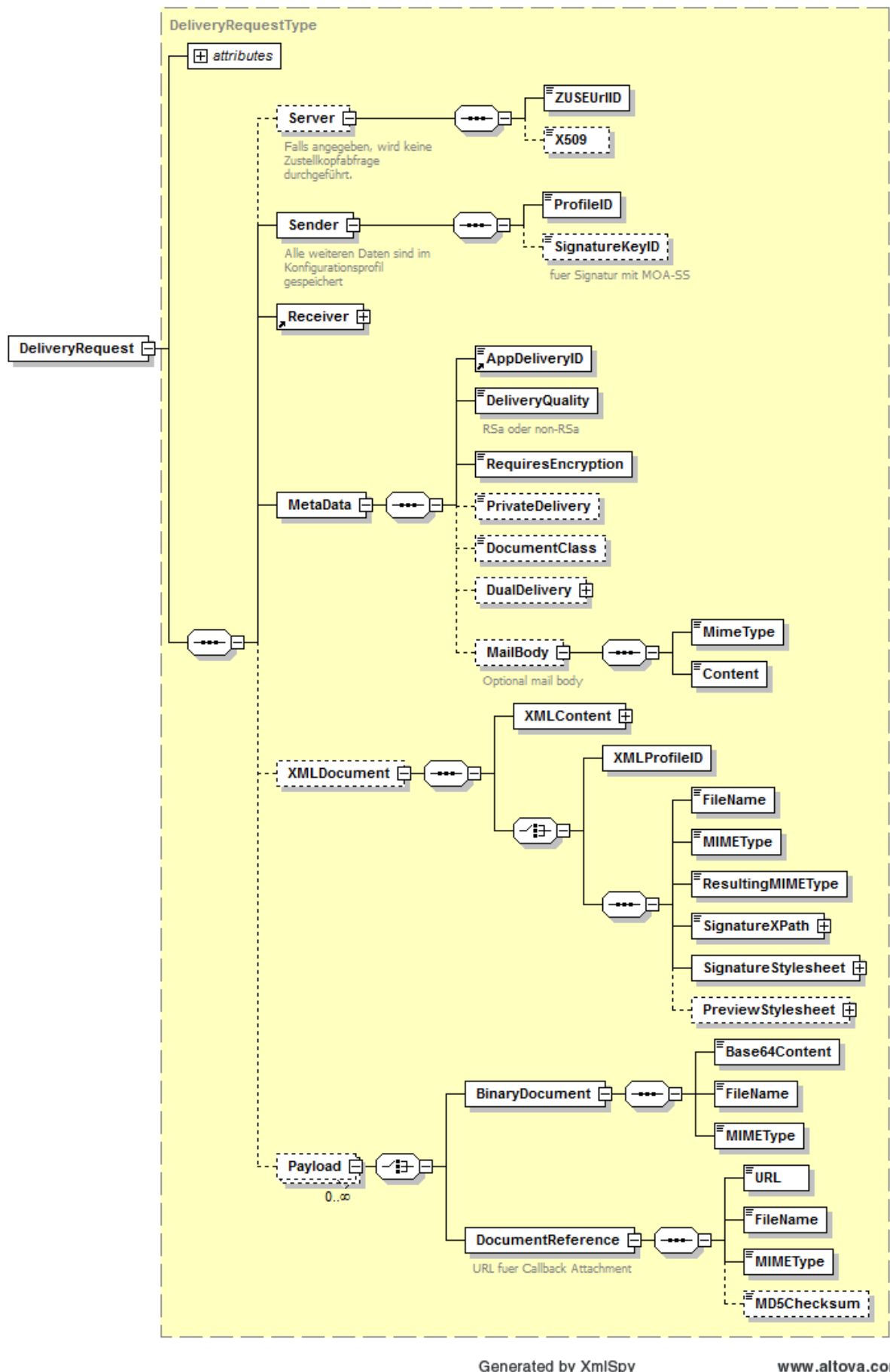


Abbildung 3.3 XML Datenformat für die Datenanlieferung

### 3.2.1 Semisynchrone Zustellung

Enthält die `DeliveryRequest` Nachricht das optionale Attribut `sync` und ist dieses auf `true` gesetzt, so wird der Request als semisynchron abgearbeitet.

In der asynchronen Abarbeitung erhält die Applikation von MOA-ZS als Antwort auf die `DeliveryRequest` Nachricht lediglich eine Übernahmebestätigung, eine Antwort bezüglich einer erfolgreichen oder fehlerhaften Zustellstückübergabe an den Zustellserver erfolgt über einen asynchronen Benachrichtigungsmechanismus (z.B. E-mail oder Webservice).

Die semisynchrone Zustellung hingegen wartet für einen bestimmten Zeitraum (konfigurierbar in der MOA-ZS Konfiguration) auf eine Statusänderung bezüglich der Abarbeitung im Backend von MOA-ZS und gibt das Resultat direkt als Antwort auf die `DeliveryRequest` Nachricht zurück. D.h. falls in diesem Zeitraum bereits ein finaler Status (Erfolg oder Fehler) vorliegt, entfällt die asynchrone Benachrichtigung und die Applikation braucht diese nicht mehr gesondert zu empfangen und zu bearbeiten.

Es gibt für die semisynchrone Zustellung folgende drei unterschiedliche Antwortmöglichkeiten auf eine `DeliveryRequest` Nachricht, welche allesamt dem gleichen XML Schema wie die asynchronen MOA-ZS `DeliveryNotification` Nachrichten folgen (siehe Abschnitt 3.4):

1. **Success** - das Zustellstück wurde erfolgreich dem Zustellserver übergeben. Die asynchrone Statusübermittlung entfällt.
2. **Error** - es ist ein Fehler beim Abarbeiten oder Übergeben des Zustellstücks aufgetreten. Die asynchrone Statusübermittlung entfällt.
3. **DeliveryStatement** - innerhalb des konfigurierten Wartezeitraums konnte kein finaler Status (Success bzw. Error) des Zustellstücks ermittelt werden. Der finale Status wird asynchron übermittelt.

Wie aus Punkt 3 ersichtlich ist, kann eine asynchrone Benachrichtigung nicht gänzlich ausgeschlossen werden und muss daher zusätzlich zum synchronen Mechanismus zur Verfügung stehen. Dies ist bspw. der Fall, wenn ein Zustellserver für den Wartezeitraum auf den Status nicht verfügbar ist und das Zustellstück erst zu einem späteren Zeitpunkt übergeben werden kann. Der finale Status der Zustellstückübergabe steht somit erst zu diesem späteren Zeitpunkt fest und wird erst dann asynchron (z.B. via E-mail oder Webservice) übermittelt.

### 3.2.2 Zustellserver (optional)

Optional kann ein `Server` Element mit der URL des Zustellservers (`zuseurlid`) und allenfalls einem hinterlegtem X.509 Zertifikat (`x509`) angegebenen werden. In diesem Fall wird keine Zustellkopfabfrage durchgeführt, sondern direkt das Zustellstück dem angegebenen Zustellserver übergeben.

Dieses Feature setzt voraus, dass bereits zuvor eine Abfrage am Zustellkopf durchgeführt wurde und das Ergebnis der Zustellkopfantwort ausgewertet wurde (auch gegen vorhandene MIME-Types usw.).

MOA-ZS kann in diesem Modus mit lokal gespeicherten Empfängerinformationen arbeiten, die z.B. über die Zustellkopf Bulkschnittstelle erstellt wurden.

### 3.2.3 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 8.1 beschrieben sind. Zusätzlich muss eine Signature-Key-Identifier (`SignatureKeyID`) für MOA-SS (Signaturserver) angegeben werden (siehe Kapitel 5).

### 3.2.4 Empfänger (Receiver)

Der Empfänger eines Zustellstücks kann auf folgende Arten adressiert werden:

- unverschlüsselte Zustell-bPK des Empfängers
- verschlüsselte Zustell-bPK des Empfängers
- bPK des Empfängers in dem Verfahrensbereich der aufrufenden Applikation + Name + Geburtsdatum
- Name + Verständigungsadresse (+ Geburtsdatum)
- Name + Adresse der Abgabestelle (+ Geburtsdatum)
- Stammzahl (= Firmenbuchnummer, Vereinsnummer, . . . ) für juristische Personen, die mit der unverschlüsselten Zustell bPK gleichzusetzen ist

Die Empfängerinformation wird im PersonData 2.0 Format (siehe [PERSONDATA]) übermittelt. Um die Handhabung der PersonData-Datenstruktur zu vereinfachen, kommt eine kompakte Version der PersonData zum Einsatz (`mzs mypersondata.xsd`, [PDCOMP]). Abbildung 3.4 zeigt den inneren Aufbau des Receiver-Elements.

Tabelle 3.1 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.

| bPK  | Inhalt des Type Elements        |
|--|---------------------------------|
| vZbPK  | urn:publicid:gv.at:ecdid+ZU     |
| ZbPK   | urn:publicid:gv.at:cdid+ZU      |
| bPK (and. Bereich)                               | urn:publicid:gv.at:cdid+Bereich |
| Firmenbuch                                       | urn:publicid:gv.at:baseid+XFN   |
| Vereinsregister                                  | urn:publicid:gv.at:baseid+XZVR  |
| Ergänzungsregister<br>für sonstige<br>Betroffene | urn:publicid:gv.at:baseid+XERSB |

Tabelle 3.1: bPKs und ihre Abbildung im Element Identification/Type

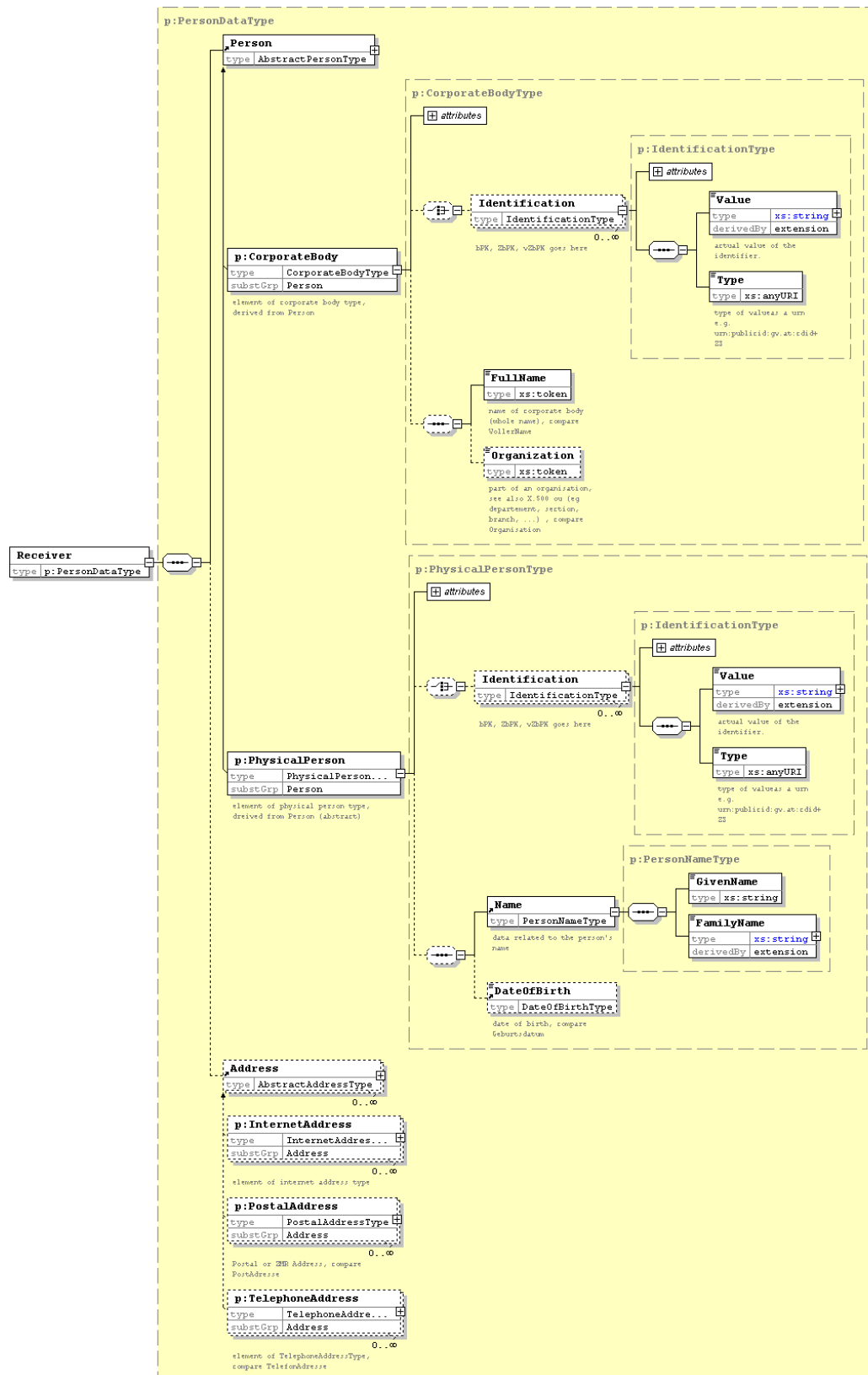


Abbildung 3.4 XML Datenformat für die Datenanlieferung: Angabe des Empfängers

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. Für nicht-natürliche Personen

(`CorporateBody`) ist das Unterelement `FullName` auszufüllen. Optional kann auch eine Untereinheit der nicht-natürlichen Person angegeben werden (`Organization`).

Wenn keine bPK als Identification-Element angeführt ist, dann muss mindestens eine Adresse (Abgabestelle, Internet oder Telefon) angegeben werden. Es können beliebig viele Adressen angegeben werden. Ein Praxisbeispiel für den Gebrauch von Name, Adresse, etc findet sich in A.1 Beispiel: Anlieferung eines Zustellstücks.

### 3.2.5 Zustell-Metainformation

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

- Zustell-ID (`AppDeliveryID`)
- Zustellqualität (`DeliveryQuality`)
  - Behördliche Zustellung:
    - RSa, RSa+
    - nonRSa, nonRSa+
    - RSb, RSb+
  - Privatzustellung:
    - R, R+
    - RS, RS+
- Verschlüsselungspflicht (boolean) (`RequiresEncryption`)

Die Zustell-ID (`AppDeliveryID`) gibt an, unter welcher Kennzahl das Zustellstück der Applikation bekannt ist (bspw. Geschäftszahl). Bei einer Zustellung mit der Qualität RSa 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 öffentlichen Schlüssel (Public-Key) für eine Verschlüsselung bekannt gegeben hat. Ist dies nicht der Fall, so ist die Anfrage an MOA-ZS mit einem entsprechenden Fehler zu quittieren.

Folgende weitere Metainformationen sind optional:

- MailBody (`MailBody`)
- Privatzustellung (`PrivateDelivery`)
- Duale Zustellung (`DualDelivery`)

Ist das optionale Element `MailBody` angegeben, so kann mit diesem Element ein individueller Inhalt des E-mails konfiguriert werden, welches als Information bei der Weiterleitung von Zustellstücken an den Empfänger angezeigt wird. Wird dieses Element verwendet, so müssen sowohl das Element `MimeType` (Dateityp) sowie das `Content` Element angegeben werden. Das `Content` Element muss den Base64-kodierten Body enthalten.

Ist das optionale Element `PrivateDelivery` im Request enthalten, so markiert dies eine nachweisliche Zusendung im Auftrag von Privaten gemäß dem Zustellgesetz (siehe [ZUSTG]). Weiterführende Informationen zur Spezifikation der privaten Zustellung siehe [ZUSEPRIV].

Ist das optionale Element `DualDelivery` im Request enthalten, so ist im Falle der elektronischen Nicht-Adressierbarkeit eine duale Zustellung auf einen im Request über das optionale Element `DualDeliveryServer` angegebenen dualen Zustellserver möglich.

Ist das Element `DualDeliveryServer` nicht angegeben, so wird ein beliebiger vorkonfigurierter dualer Zustellserver aus der MOA-ZS Konfiguration herangezogen. Ist das Element vorhanden, so kann dieses entweder die absolute URL des dualen Zustellservers enthalten

oder einen Bezeichner (friendlyname), der auf einen vorkonfigurierten dualen Zustellserver in der Konfiguration hinweist.

Weiterführende Informationen zur Spezifikation der dualen Zustellung siehe [ZUSEMOD] und [ZUSEMSG].

### 3.2.6 Zustellstück – Payload

Ein Zustellstück kann aus einer oder mehreren XML oder binären Dateien bestehen. XML Dateien werden direkt über `XMLDocument/XMLContent` eingebunden. Zusätzlich können ein Signatur-Stylesheet (`SignatureStylesheet`) und ein Stylesheet für die Vorschau im Browser (`PreviewStylesheet`) übergeben werden. Wird kein Stylesheet übergeben, so werden Stylesheets, welche im Profil konfiguriert wurden, benutzt.

Binäre Dateien werden entweder im `BinaryDocument` als Base64-kodierter Inhalt (`Base64Content`) eingebunden, oder als Callback-Attachment unter `DocumentReference` angesprochen. Für Base64-kodierte 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 (siehe [HTTP11]) unterstützen. Für eine externe Datei kann zusätzlich eine MD5 Prüfsumme (siehe [MD5]) angegeben werden.

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

Es gibt zwei Arten von Erfolgsmeldung:

- Übernahmebestätigung/Empfänger adressierbar
- Partielle Übernahmenbestätigung (Callback option)

und eine Art Fehlermeldung, die durch einen Fehlercode und eine Fehlernachricht näher spezifiziert wird. Details sind der Abbildung 3.5, dem Beispiel 3.1, Beispiel 3.2 und Beispiel 3.3, sowie der Tabelle 3.2 zu entnehmen.

Fehler 521 “No public key available” tritt nur auf, wenn die Applikation bei der Übergabe des Zustellstücks 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 (sehr unwahrscheinlich).

Außer diesen Fehlern gibt es noch automatisch generierte Fehlermeldungen des HTTP Servers (z.B. 500 - Internal Server Error) und des SOAP Servers (SOAP Faults über falsches Datenformat etc.). Eine Liste dieser Fehlercodes ist der Dokumentation von MOA-ZS beizufügen.

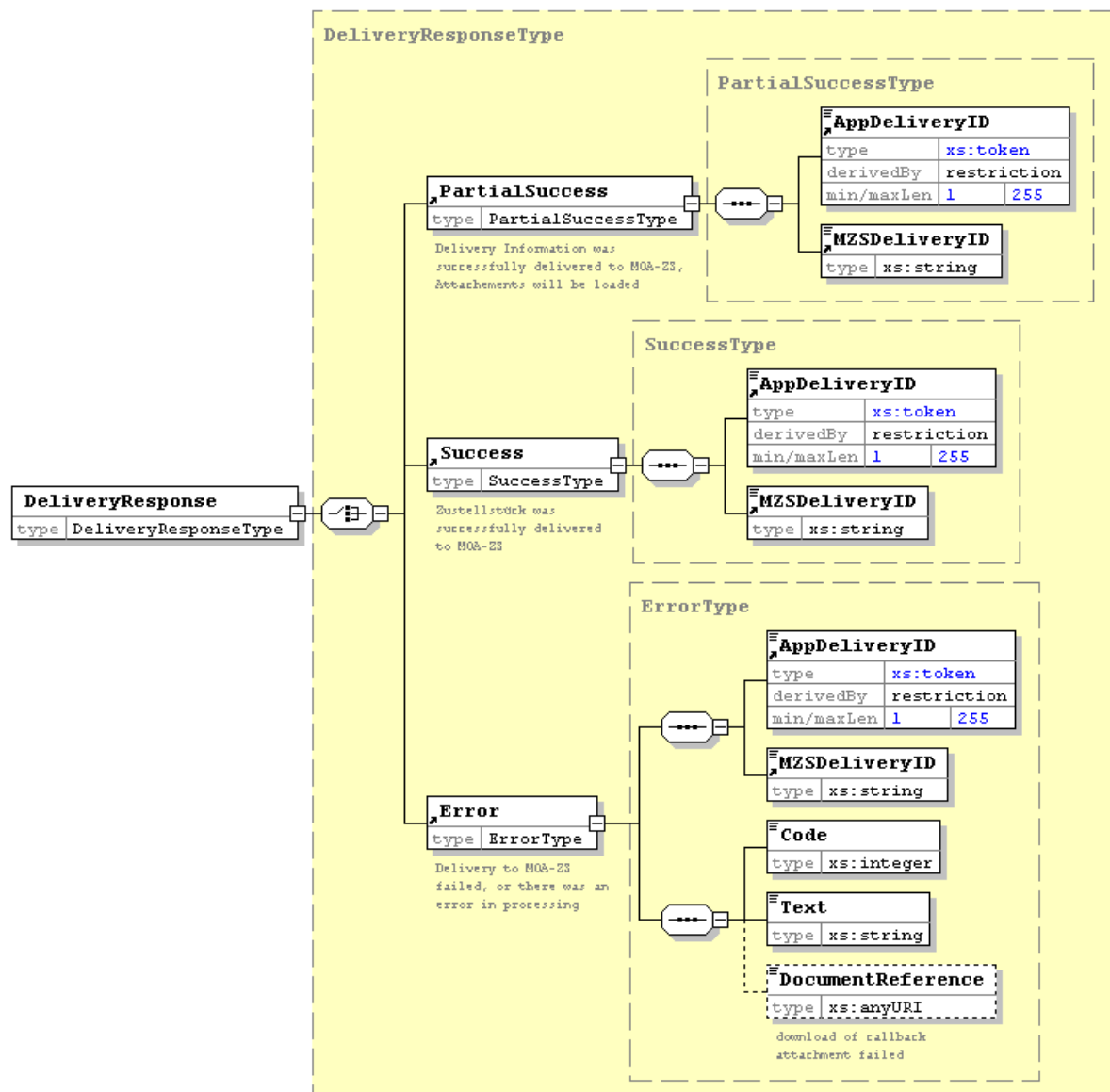


Abbildung 3.5 XML Datenformat für die Antwort auf Datenanlieferung (synchrone Kommunikation): Success, PartialSuccess und Error.

```
<?xml version="1.0" encoding="UTF-8"?>
<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:MZSDeliveryID>
      </mzs:Success>
    </mzs:DeliveryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Beispiel 3.1: Nachrichtenformat für Erfolgsmeldungen, Übernahmenbestätigung, Bestätigung der Adressierbarkeit

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<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:AppDeliveryID>
        <mzs:MZSDeliveryID>abc765432</mzs:MZSDeliveryID>
      </mzs:PartialSuccess>
    </mzs:DeliveryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

### Beispiel 3.2: Nachrichtenformat für Erfolgsmeldungen – partielle Übernahmenbestätigung im Callback Fall

```
<?xml version="1.0" encoding="UTF-8"?>
<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:MZSDeliveryID>
        <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 3.3: Nachrichtenformat für Fehlermeldungen und Abbruchsverständigungen

| Code  | Fehlermeldung                         | 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                      |                 |
| 501   | Missing metainformation               |                 |
| 502   | Attachment could not be loaded        | Attachment name |
| 503   | Attachment MD5 verification failed    | Attachment name |
| 504   | MOAZS internal server error           |                 |
| Serverfehler – SZR (Stammzahlenregister)        |                                       |                 |
| 510   | SZR is not responding                 |                 |
| 511   | bPK not valid                         |                 |
| 512   | bPK-Domain not valid                  |                 |

| Code                           | Fehlermeldung                         | File |
|--------------------------------|---------------------------------------|------|
| 513                            | bPK unresolvable                      |      |
| 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 3.2: Fehlercodes in der Zustellstückannahme

### 3.4 Datenformate der Benachrichtigungen (DeliveryNotification)

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

`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 3.2 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`).

Die Übermittlung von `DeliveryNotification` ist nicht zwingend erforderlich und kann über das jeweilige MOA-ZS Profil unterdrückt werden. Die Nachricht `DeliveryNotificationACK` dient nur dazu den Erhalt der `DeliveryNotification` Nachrichten zu bestätigen und enthält keine neue Information.

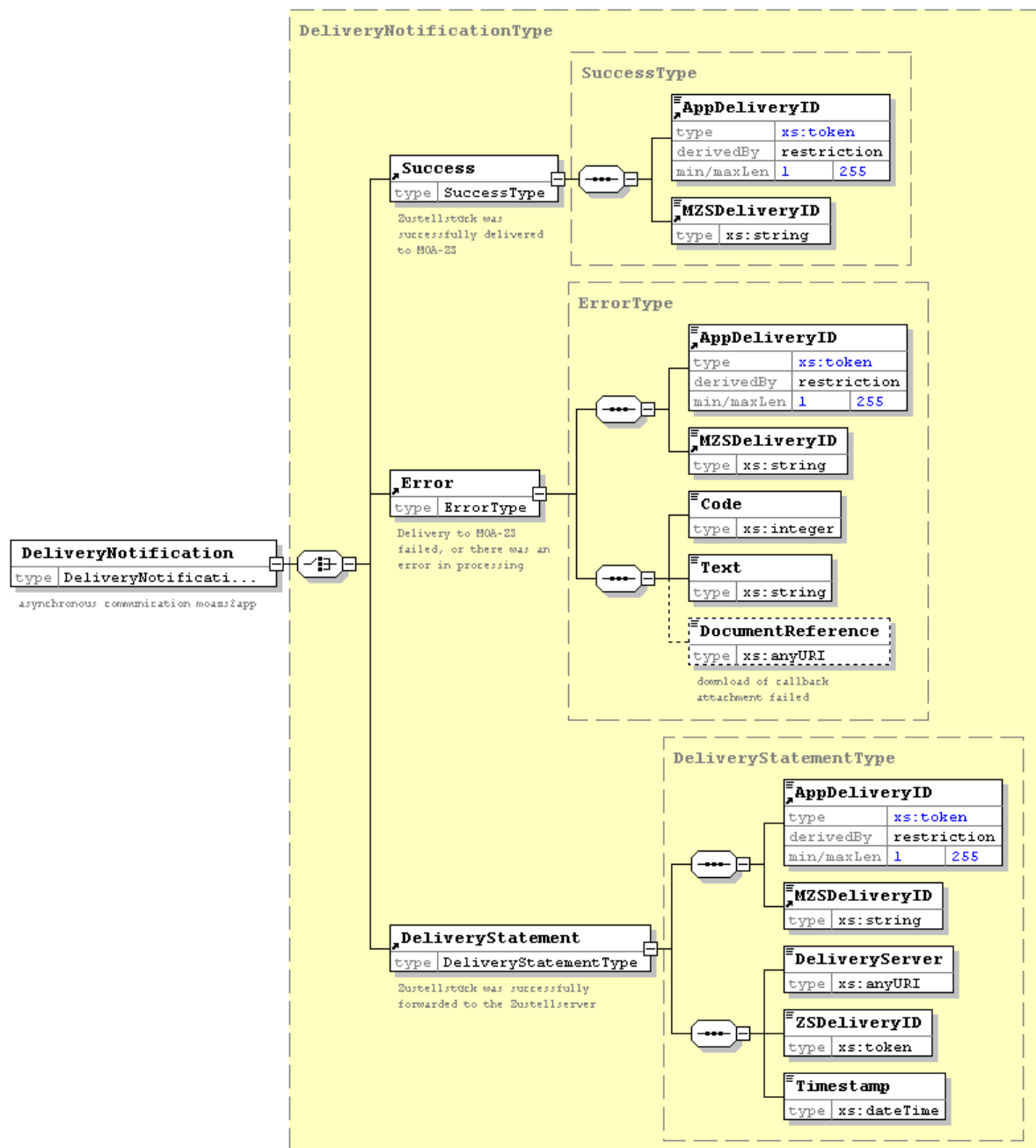


Abbildung 3.6 XML Datenformat für die Verständigungen zwischen MOAZS und Applikation (asynchrone Kommunikation): Success, Error und DeliveryStatement.

### 3.5 Vollständigkeitsprüfung

Vor der Adressierbarkeitsprüfung muss die Vollständigkeit der Zustell-Angaben geprüft werden. Unvollständige Zustellanträge sind mit dementsprechender Fehlermeldung (Error 501, siehe Abschnitt 3.3) abzuweisen. Die notwendigen Zustell-Angaben können Abbildung 3.3 und Abbildung 3.4 entnommen werden. Verpflichtende Angaben sind mit durchgehender Linie umrahmt, optionale Angaben mit strichlierter.

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



## 4 Prüfung der Adressierbarkeit

Die Prüfung der Adressierbarkeit erfolgt in zwei Schritten:

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

### 4.1 bPK Umrechnung beim SZR

Die Kommunikation mit dem SZR erfolgt gemäß der SZR Spezifikation [SZRSPEC].

#### 4.1.1 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 (bPK) wird vom SZR in ein für die Zustellung geeignetes Personenkennzeichen umgewandelt, und mit dem öffentlichen Schlüssel des Zustellkopfs verschlüsselt an MOA-ZS retourniert (Verschlüsseltes Zustell-bPK - vZbPK). Der öffentliche Schlüssel des Zustellkopfes wird am SZR mittels eines in der Konfiguration angegebenen Verfahrenskennzeichens (VKZ) angegeben.

#### 4.1.2 MOA-ZS Anfrage

Die MOA-ZS Anfrage wird in der Spezifikation des Stammzahl-Registers (siehe [SZRSPEC]) als bPK-Transformation betitelt.

Die für eine eindeutige Umwandlung benötigten Daten setzen sich wie folgt zusammen:

- Identifikationsdaten  
Die Identifikationsdaten, die zu einer eindeutigen Identifikation des Empfängers nötig sind, bestehen aus den folgenden Daten:
  - Familienname
  - Vorname
  - Geburtsdatum
- Input-bPK  
Das bereichsspezifische Personenkennzeichen, welches von der Fachapplikation an MOA-ZS übergeben wurde.
- Input-Bereichskennung  
Damit das Stammzahlen-Register die Umrechnung in die verschlüsselte Zustell-bPK vornehmen kann, muss der Bereich der übergebenen bPK im Request an MOA-ZS mitgeliefert werden.
- Output-Bereich  
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.

### 4.1.3 Antwort des Stammzahlenregisters

Die Antwort des Stammzahlenregisters ist das verschlüsselte Personenkennzeichen der elektronischen Zustellung, welches mit dem öffentlichen Schlüssel des Zustellkopfs verschlüsselt ist.

### 4.1.4 Adressierbarkeitsanfrage beim Zustellkopf

Wurden beim Zustellstück mehrere Empfängeradressen angegeben, so ist für jede Adresse der Zustellkopf gesondert zu befragen (siehe [EGOVG]). Die Kommunikation mit dem Zustellkopf erfolgt gemäß der Interface Spezifikation Applikation ↔ Zustellkopf (siehe [ZUSEKOPF]). Die Antwort besteht aus einer Ergebnisliste oder einer Fehlermeldung.

Wurden mehrere Anfragen gestellt, so sind die Antworten abzuwarten. Elektronische Postfächer mit hinterlegtem Schlüssel sind zu bevorzugen. Hat die Applikation in der Übergabe des Zustellstücks eine Verschlüsselungspflicht (`RequiresEncryption="true"`) angegeben, so muss eine elektronische Zustellung abgelehnt werden, falls kein elektronisches Postfach mit hinterlegtem Schlüssel gefunden werden kann (Fehler 521, Abschnitt 3.3). Bei mehreren gleichwertigen Postfächern sind jene in der Konfiguration als bevorzugte Zustellserver angegebenen zu verwenden. Konnte kein bevorzugter Zustellserver ermittelt werden, ist einer davon per Zufall auszuwählen.

Ist die elektronische Zustellung möglich, so ist das Zustellstück persistent zu speichern. Erst danach kann eine (partielle) Annahmebestätigung (`DeliveryResponse/Success` oder `PartialSuccess`) (siehe Abschnitt 3.3) an die Applikation retourniert werden.

### 4.1.5 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 3.3) an die Applikation zu retournieren. War der Download/die Überprüfung erfolgreich, so ist nach erfolgter persistenter Speicherung eine Übernahmebestätigung zu retournieren.

## 5 Anbringen der Absendersignatur

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

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

### 5.1 Funktionaler Überblick

Diese Schnittstellenbeschreibung 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 (siehe [XMLDSIG]) als Enveloped Signature ausgeführt. Enthält das Zustellstück ein XML Dokument, so kann eine Signatur an diesem Dokument angebracht werden. Bei mehreren XML Dokumenten kann die Signatur nur im jeweils ersten Dokument (Reihenfolge in der Payload bei der Anlieferung) angebracht werden. Besteht die Zustellung ausschließlich aus binären Dokumenten, so kann ein XML Deckblatt erstellt werden, in das die Signatur anschließend 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.

### 5.2 Datenaufbereitung

Bei der Datenaufbereitung sind folgende Fälle zu unterscheiden:

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

Zusätzlich ist der `SignatureKeyID` aus dem Request zu ermitteln. Dieser wird als `KeyIdentifier` zur Identifizierung des zu verwendenden Signaturschlüssels an MOA-SS übermittelt.

Wurde kein `SignatureKeyID` angegeben, so wird eine Erstellung der Signatur nicht durchgeführt.

#### 5.2.1 XML Nachrichten

Jedes XML Zustellstück kann prinzipiell über 2 XSLT Stylesheets verfügen:

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

Die Stylesheets werden entweder im Request mit übergeben, oder durch die `XMLProfileID` in der Konfiguration referenziert.

Das Signatur-Stylesheet wird als `dsig:Transform` Inhalt an MOA-SS übergeben. Das Vorschau-Stylesheet wird nicht zur Signatur an MOA-SS übergeben, da dieses nur zur Anzeige am Zustellserver benötigt wird.

## 5.2.2 Binäre Zustellstücke

Besteht das Zustellstück ausschließlich aus binären Dateien, so kann optional zusätzlich ein 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 5.1).

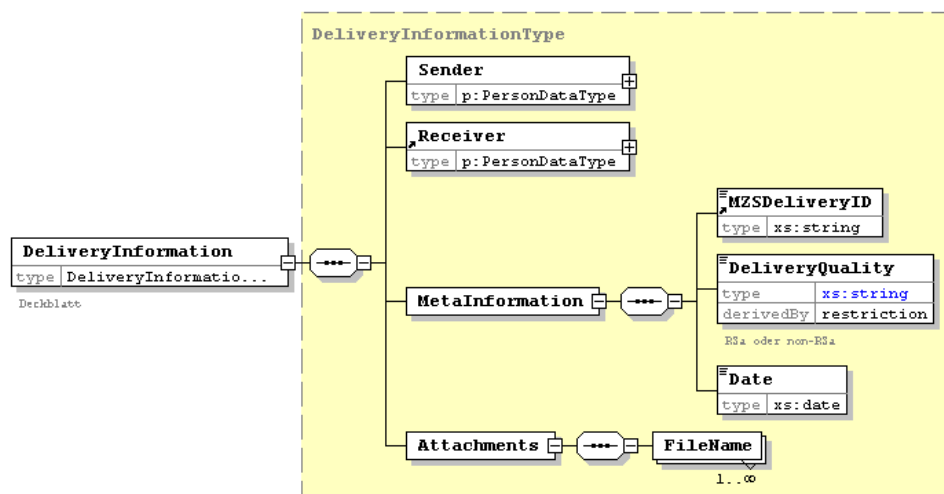


Abbildung 5.1: XML Format für das Attachment Deckblatt

```

<?xml version="1.0" encoding="UTF-8"?>
<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/namespaces/moa/20020822#">
      ...
    </moa:CreateXMLSignatureRequest>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
  
```

Beispiel 5.1: SOAP Format der Signaturanfrage

## 5.2.3 XML und binäre 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.

## 5.3 MOA-SS Schnittstelle: Request

MOA-SS wird über seine SOAP Schnittstelle (siehe [MOASPS]) angesprochen (siehe auch Beispiel 5.1). Dabei kommen in diesem Fall ausschließlich die Nachrichten `CreateXMLSignatureRequest` (Anfrage) und `CreateXMLSignatureResponse` (Antwort) gemäß des MOA-SS Interface Schemas (siehe [MOASPS]) zum Einsatz.

### 5.3.1 CreateXMLSignatureRequest

Abbildung 5.2 zeigt eine Übersicht über das Request Nachrichtenformat.

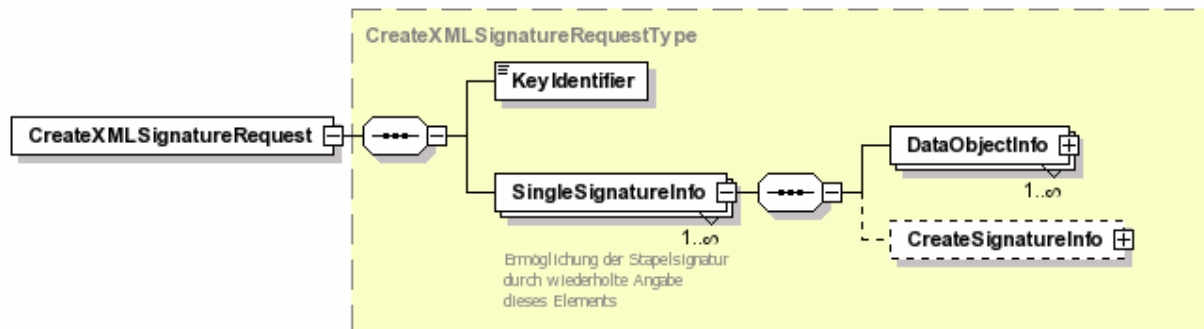


Abbildung 5.2: XML Format für MOA-SS Request (high-level view)

Das Element `CreateXMLSignatureRequest` umschließt die gesamte Anfrage. `KeyIdentifier` gibt den zu verwendenden Signatur-Schlüssel an und liegt bereits als `SignatureKeyID` in der ursprünglichen MOA-ZS Anfrage vor. Das Element `SingleSignatureInfo` umschließt wiederum alle zu signierenden Teile des Zustellstücks und einen Block mit Signatur Metainformation. Im Falle der Zustellung kommt also nur genau ein `SingleSignatureInfo` Element in der SOAP Nachricht vor. (`CreateSignatureInfo`, Abschnitt 5.3.2).

Die Teile des Zustellstücks werden in `DataObjectInfo` Elemente gekapselt. Die genaue Vorgangsweise ist abhängig vom Typ des Zustellstückteils und wird in Abschnitt 5.3.3 genauer behandelt.

### 5.3.2 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 5.3).

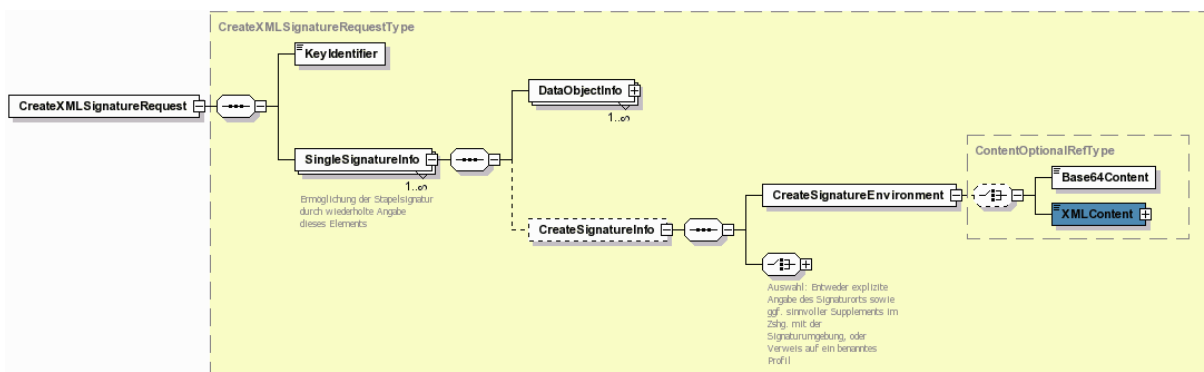


Abbildung 5.3: Das XML Dokument welches die Signatur aufnehmen soll, wird innerhalb von `XMLContent` eingefügt.

Die genaue Position der Signatur wird über `CreateSignatureLocation` unter `//SingleSignatureInfo/CreateSignatureInfo/CreateSignatureEnvironmentProfile` eingefügt (siehe Abbildung 5.4). `CreateSignatureLocation` gibt dabei das Parentelement an. Das `Index` Attribut in `CreateSignatureLocation` gibt an, an welcher Stelle innerhalb des Parentelements die Signatur eingefügt werden soll. Dabei bedeutet `Index="n"`

- $n=-1$  als letztes Childelement
- $n=0$ : als erstes Childelement
- $n \geq 1$ : nach dem  $n$ -ten Childelement

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

Beispiel 5.2: Erläuterung von CreateSignaturLocation und seines Index Attributs

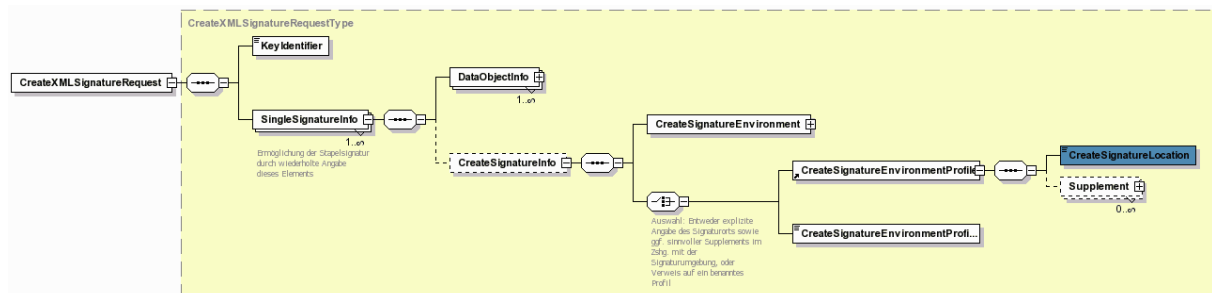


Abbildung 5.4: Die Position, an der die Signatur in das bestehende XML-Dokument eingefügt werden soll, wird über das Element `CreateSignatureLocation` angegeben. Feinpositionierung erfolgt über das Attribut `Index`.

### 5.3.3 DataObjectInfo

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

- XML Dokument, das als Envelope für die Signatur dient
- Weitere XML Dokumente
- Binäre Dateien

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 5.5). Für jedes Dokument wird ein eigenes `DataObjectInfo` Element angelegt.

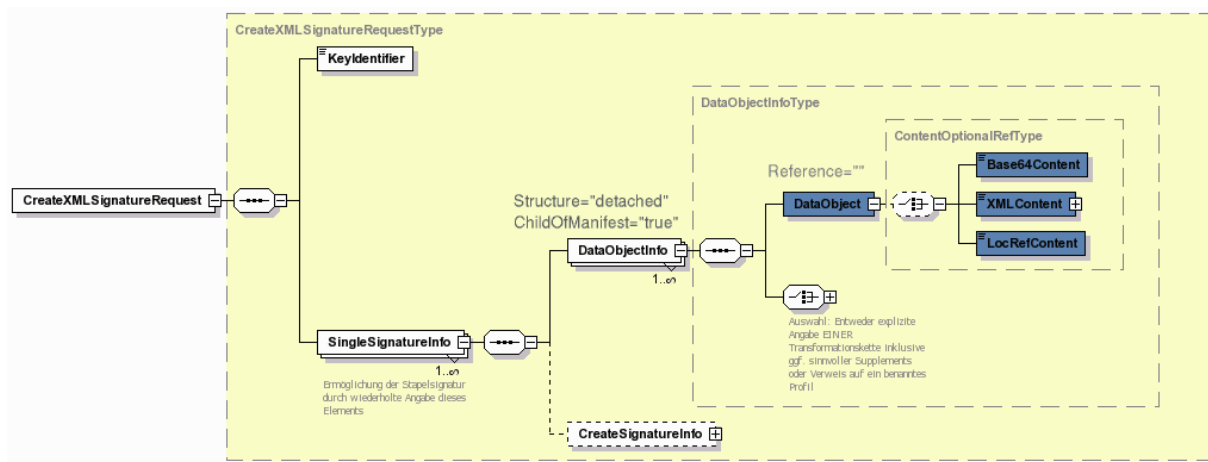


Abbildung 5.5: Zustellstückteile können als `Base64Content`, als `XMLContent` oder als `LocRefContent` eingebunden werden.

### 5.3.3.1 XML Envelope

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

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 5.6 zeigt die Eingliederung von Transformationen. Beispiel 5.3 gibt an, wie die Transformation die den Signaturblock von der Signatur ausnimmt, auszusehen hat. Die angewandte Transformation ist ein Standardtransformation aus der XMLDSIG Spezifikation (siehe [XMLDSIG]).

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

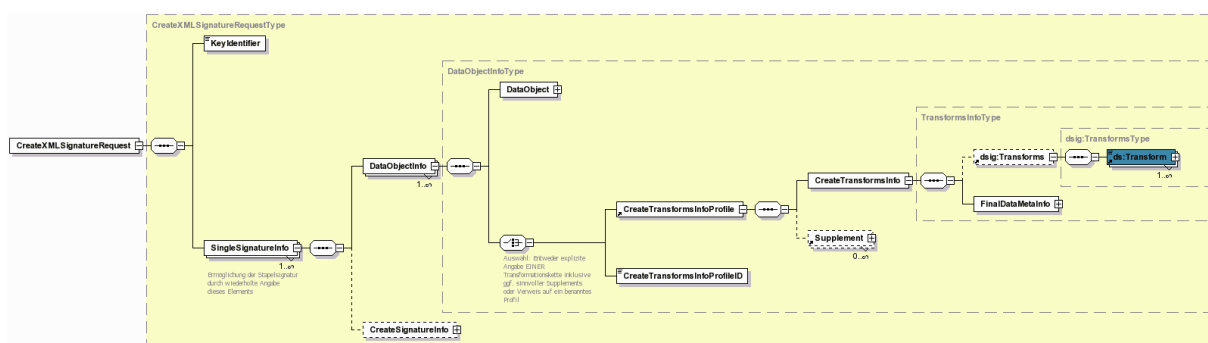


Abbildung 5.6: Es können mehrere Transformationen angegeben werden. Alle Transformationen werden der angegebenen Reihenfolge nach ausgeführt und das endgültige Ergebnis wird signiert.

```
<CreateTransformsInfo>
  <dsig:Transforms>
    <dsig:Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
    <dsig:Transform Algorithm="http://www.w3.org/TR/1999/REC-xslt-
199991116">
      <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 5.3: Die erste Transformation nimmt den Signaturblock von der Signatur aus, die zweite beinhaltet das Stylesheet für die Anzeige in der Bürgerkartenumgebung.

### 5.3.3.2 XML Dokumente

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 5.3.3 im `Transform` Element eingebettet.

### 5.3.3.3 Binäre Dokumente

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

```

<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 5.4: SOAP Format einer Signaturantwort.

## 5.4 MOA-SS Interface: Response

Das Antwortformat ist recht einfach gehalten: Das signierte XML Dokument befindet sich im `SignatureEnvironment` Element, Fehler im `ErrorResponse` (siehe Abbildung 5.7). Dabei ist zu beachten, dass nur jene Dokumente inkludiert sind, die beim Request mit `DataObjectInfo@Structure="enveloping"` angegeben wurden.

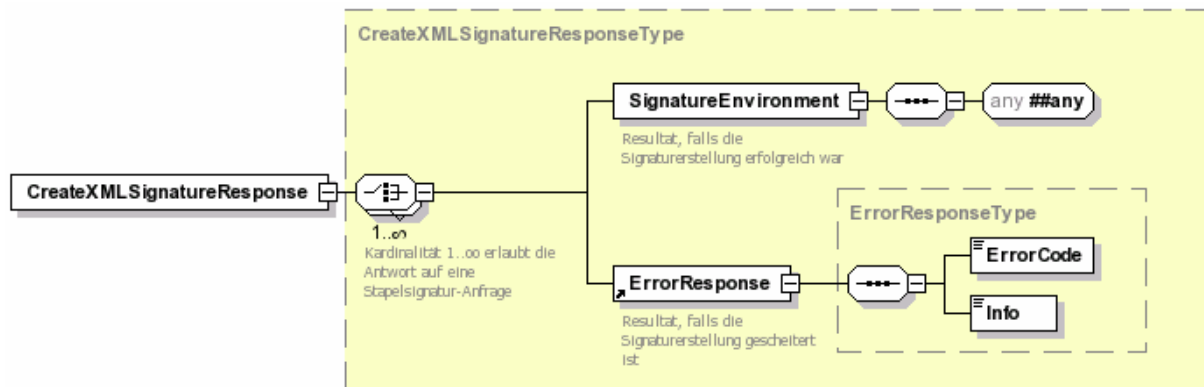


Abbildung 5.7: Im Erfolgsfall befindet sich das signierte XML Dokument im SignatureEnvironment Element, Fehler werden als ErrorResponse übermittelt.

```
<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 5.5: SOAP Format eines Signaturfehlers

## 6 Verschlüsselung des Zustellstücks

Hat der Bürger bei einem Zustelldienst einen Public-Key für die Verschlüsselung im X.509 DER Format (siehe [X509]) 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 abzubrechen und die Applikation diesbezüglich zu verständigen (Fehler 540-542, Abschnitt 3.3).

### 6.1 Zustellstückaufbereitung

Bevor die eigentliche Verschlüsselung stattfinden kann, muss das Zustellstück entsprechend aufbereitet werden. Es sind zwei Methoden zu implementieren (siehe Abbildung 6.1):

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

Die jeweils zu benutzende Methode ist über das Konfigurationsfile zu ermitteln (siehe Abschnitt 8.3).

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, usw.) in einem MIME-Container zusammenzufassen.

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 elektronische 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 zu Recht 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.

Beispiel 6.1 zeigt den MIME Envelope eines Zustellstücks ohne Zustellcontainer.

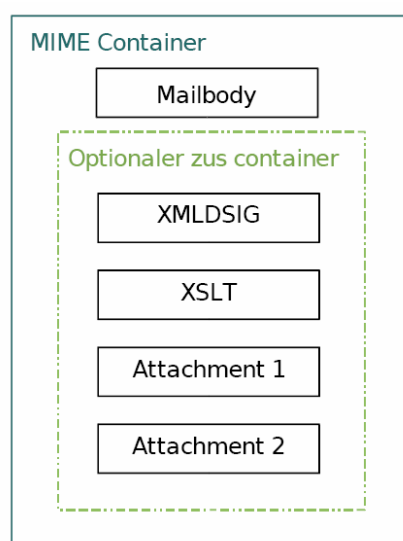


Abbildung 6.1: Zustellstück Container: MIME, ZUS

Das .ZUS File besteht aus einem ZIP-Container (siehe [ZIP]) 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.

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"
JVBERi0xLjQNCiXk9tzfDQoxIDAgb2JqDQo8PCAvTG VuZ3RoIDIgMCBSDQogICAvRmlsdGVyIC9
G
bGF0ZURlY29kZQ0KPj4NCnN0cmVhbQ0KeJyVVNtqgwzAMfQ/4H/RcmGf5GkMZNF37XgjsB7ZujG3
Q
...
```

#### Beispiel 6.1: 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 6.2: MIME Header für den Transfer von .ZUS Containern

## 6.2 Verschlüsselung

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

Abbildung 6.2 zeigt ein verschlüsseltes Zustellstück und Beispiel 6.3 zeigt den MIME-Envelope für den Transfer von verschlüsselten Zustellstücken.

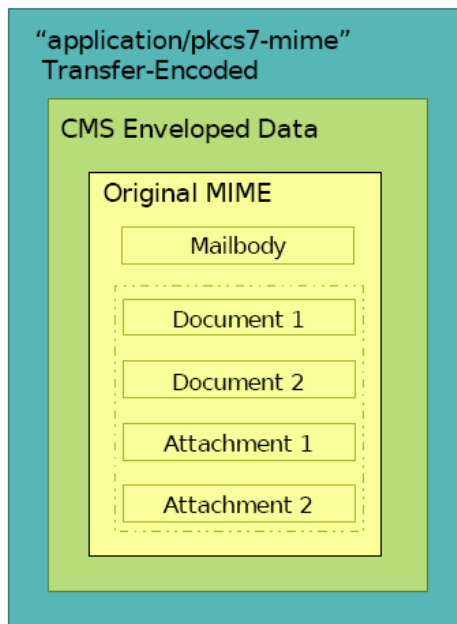


Abbildung 6.2: Zustellstück Container: CMS, S/MIME

```
MIME-Version: 1.0
Content-Type: application/x-pkcs7-mime; name="smime.p7m"
Content-Transfer-Encoding: base64
```

```
MIMC7VMGCSqGSib3DQEHA6CDAu1DMIMC7T4CAQAxggEuMIIBKgIBADCBkjCBizEL
MAkGA1UEBhMCQVQxSDBGBgNVBAoTP0EtVHJlc3QgR2VzLiBmLiBTaWN0ZXJoZWl0
iBTaWN0ZXJoZWl0.....
```

Beispiel 6.3:: MIME Header für den Transfer von CMS Containern

## 7 Übergabe an einen Zustelldienst

Die Übergabe an den Zustelldienst erfolgt gemäß der Interface Spezifikation Applikation ↔ ZUSE (siehe [ZUSEMSG]).

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

## 8 Konfiguration

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

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.

### 8.1 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 ProfileID abgerufen.

Folgende Information sind zu erfassen:

- ProfileID
- Endpoint für Benachrichtigungen (Webservice, Postfach, e-mail)
- Administrator (Name, e-mail, tel)
- Absenderinformation
  - Name der Behörde/Körperschaft (PersonData)
  - OID/VKZ der 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)
- Verständigung über die Übergabe an Zustelldienst (DeliveryNotification ja/nein)
- Maximale Wartezeit für die synchrone Zustellung

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:

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

### 8.2 Komponenten

Pro extern aufzurufende Komponente (SZR, Zustellkopf, MOA-SS, Zustelldienste) kann folgendes konfiguriert werden

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

### 8.3 MOA-ZS Interne Konfiguration

- Zustellcontainerformat (MIME oder ZUS)
- "Queue congestion" Parametriesierung nach Größe und/oder absoluter Zustellstückzahl
- Bevorzugte Zustellserver sortiert nach Prioritäten

## 9 Logging

Folgende Informationen müssen für jeden elektronischen Zustellversuch erfasst werden

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

Im Falle der erfolgreichen Übergabe an den Zustelldienst

- MZSID
- ZustelldienstID
- Ausgangstimestamp

Im Falle eines Abbruchs

- MZSID
- Fehlercode
- Abbruchtimestamp

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

- KomponentenID
- Fehlertimestamp
- Fehlercode
- Fehlerklasse
- 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:dsig="http://www.w3.org/2000/09/xmldsig#">
  <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>
    <p:InternetAddress>
      <p:Address>hermann.maier@superg.com</p:Address>
    </p:InternetAddress>
  </Receiver>
  <MetaData>
    <AppDeliveryID>4711</AppDeliveryID>
    <DeliveryQuality>RSa</DeliveryQuality>
    <RequiresEncryption>true</RequiresEncryption>
  </MetaData>
  <XMLDocument>
    <XMLContent>
      <!-- Hier steht ein XML Bescheide -->
    </XMLContent>
    <XMLProfileID>123456</XMLProfileID>
  </XMLDocument>
  <Payload>
    <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

## Referenzen

|              |  |
|--------------|--|
| [XMLDSIG]    | Mark Bartel, XML Signature Syntax and Processing.<br><a href="http://www.w3.org/TR/xmlsig-core/">http://www.w3.org/TR/xmlsig-core/</a> , 2002.   |
| [X509]       | S. Chokhani. RFC 3647 –Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework.<br><a href="http://www.ietf.org/rfc/rfc3647.txt">http://www.ietf.org/rfc/rfc3647.txt</a> , 2003. |
| [HTTP11]     | R. Fielding and et. al. RFC 2616 – Hypertext Transfer Protocol – HTTP/1.1. <a href="http://www.ietf.org/rfc/rfc2616.txt">http://www.ietf.org/rfc/rfc2616.txt</a> , 1999.   |
| [CMS]        | R. Housley. RFC 3369, Cryptographic Message Syntax   |
| [RSA]        | B. Kaliski. RFC2437 – PKCS #1: RSA Cryptography Specifications.<br><a href="http://www.ietf.org/rfc/rfc2437.txt">http://www.ietf.org/rfc/rfc2437.txt</a> , 1998.   |
| [ZUSELDAP]   | Tauber A., Reichstädter P., Hörbe R., Zustellung LDAP Schemabeschreibung, März 2008  |
| [SMIME]      | S. Dusse. RFC 2311 – S/MIME Version 2 Message Specification.<br><a href="http://www.ietf.org/rfc/rfc2311.txt">http://www.ietf.org/rfc/rfc2311.txt</a> , 2311.  |
| [MIME]       | N Freed and N. Borenstein. RFC 2045 – Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, November 1996.   |
| [LDAPSEARCH] | T. Howes. RFC 2254 – The String Representation of LDAP Search Filters, 1997.   |
| [RSA]        | J. Jonsson and B. Kaliski. RFC 3447 – Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1 , 2003.   |
| [URL]        | T. Berners Lee. RFC 1738 – Uniform Resource Locators (URL), 1994.  |
| [ZUSEKOPF]   | Tauber A., Rössler T., Elektronische Zustellung - Zustellkopf Schnittstellenspezifikation, März 2008   |
| [ZUSEMOD]    | Reichstädter P., Tauber A., Elektronische Zustellung – Modell und Prozesse, März 2008.   |
| [ZUSEMSG]    | Rössler T., Tauber A., Reichstädter P., Zustellung Message Spezifikation (Applikation an Zustellserver), März 2008   |
| [PERSONDATA] | Larissa Naber. Persondata Schema 2.0.<br><a href="http://reference.egovernment.gv.at/XML-Struktur-für-Personendaten.306.0.html">http://reference.egovernment.gv.at/XML-Struktur-für-Personendaten.306.0.html</a> , 2004.   |
| [PDCOMP]     | PersonData 2.0 Blueprint for app2mzs interface, mzs_mypersondata_en.xsd, 2004.   |
| [SAML]       | OASIS. Saml 1.0 specification.<br><a href="http://www.oasisopen.org/committees/download.php/2290/oasis-saml-1.0.zip">http://www.oasisopen.org/committees/download.php/2290/oasis-saml-1.0.zip</a> , 2002.                  |
| [SOAP]       | W3C. Simple Object Access Protocol (SOAP) 1.1.<br><a href="http://www.w3c.org/TR/2000/NOTE-SOAP-20000508/">http://www.w3c.org/TR/2000/NOTE-SOAP-20000508/</a> , 2000.  |
| [CERTOID]    | Hollosi, Arno: X.509-Zertifikatserweiterungen für die Verwaltung. Version 1.0.3 vom 21.02.2005.  |
| [MD5]        | R. Rivest, RFC 1321, The MD5 Message-Digest Algorithm, 1992.   |

|            |   |
|------------|---|
| [SZRSPEC]  | Hörbe R., Anforderungen an das Stammzahlen-Register (SZR-N), sz2-spec V3.0.3 2006-12-20   |
| [EGOVG]    | Bundesgesetz über Regelungen zur Erleichterung des elektronischen Verkehrs mit öffentlichen Stellen (E-Government-Gesetz – E-GovG), BGBl. I Nr. 10/2004, idF. BGBl. I Nr. 7/2008. |
| [MOASPSS]  | MOA SP-SS, Spezifikation Module für Online Anwendungen – SP und SS, 1.3.0, 24.08.2005   |
| [SIGG]     | Bundesgesetzblatt für die Republik Österreich, Signaturgesetz – SigG, Bundesgesetz über elektronische Signaturen  |
| [ZIP]      | PKWare Inc, .ZIP File Format Specification.<br><a href="http://www.pkware.com/documents/casestudies/APPNOTE.TXT">http://www.pkware.com/documents/casestudies/APPNOTE.TXT</a>      |
| [ZUSTG]    | Bundesgesetz vom 1. April 1982 über die Zustellung behördlicher Schriftstücke (Zustellgesetz), BGBl. Nr. 200/1982, idF. BGBl. I Nr. 5/2008.                                       |
| [ZUSEPRIV] | Tauber A., Rössler T., Reichstädter P., Elektronische Zustellung - Nachweisliche Zusendung im Auftrag von Privaten, zusepriv-1.3.0  |