einheitliches XML-basiertes Transportverfahren

Die eXTra-Standardnachrichten
Überblick

Gesamtangebot auf Basis der Schemadatei V1.3/V1.4 der eXTra-Standardnachrichten

Version 1.4
Ausgabestand 1.4.0

Herausgeber:

AWV – Arbeitsgemeinschaft für wirtschaftliche Verwaltung e. V.
Düsseldorfer Str. 40
65760 Eschborn
Vereinsregister 73 VR 5158, Amtsgericht Frankfurt am Main
Telefon: 0 61 96/7 77 26-0
Fax: 0 61 96/7 77 26-51
Mail: info@awv-net.de
Web: www.extra-standard.de, www.awv-net.de.

Das vorliegende Dokument „Die eXTra-Standardnachrichten - Überblick“ des einheitlichen XML-basierten Transportverfahrens „eXTra“ wurde von Mitarbeiterinnen und Mitarbeitern des AWV-Arbeitskreises 2.1 „Vereinheitlichung von Datenübermittlungssystemen“ im Fachausschuss 2 „Verwaltungsvereinfachung und Entbürokratisierung im personalwirtschaftlichen Umfeld“ entwickelt.

Erstausgabe Version 1.4

Autor[en]	Datum	Beschreibung
[gelöscht]	25.08.2013	Erstausgabe Version 1.4 Dieses Dokument ersetzt das Dokument „ Neue eXTra Standardnachrichten V1.0“ Dieses Dokument beschreibt die eXTra Standardnachrichten, die mit der Schemadatei V1.3 oder V1.4 erzeugt werden können. Die Neuerungen der V1.4 gegenüber der V1.3 sind die neuen Standardnachrichten ListRequest und ListResponse sowie die erweiterte Standardnachricht DataRequest der Version V1.3. Zur Orientierung: Zum Zeitpunkt der Erstausgabe des Überblicks der Standardnachrichten lautet die Version des korrespondierenden eXTra Basis-Standards V1.3

Inhalt

1. Hinweise zu diesem Dokument

1.1. Notation

Verweise auf Stellen innerhalb dieses Dokumentes referenzieren die Absatznummer und schließen sie in runde Klammern ein (z.B. „(4)“); Verweise auf externe Dokumente haben die Form eines Kurznamens aus Großbuchstaben und stehen in eckigen Klammern, z.B. [XSD]. Eine Übersicht der Referenzen befindet sich im Anhang (7.1).

Ist ein Begriff im Glossar (7.2) erläutert, so wird er bei der ersten Verwendung im Dokument mit einem vorangestellten „ⓘ“ versehen, z.B. „ⓘ Plug-In“.

Hervorhebungen sind *kursiv* gesetzt.

1.2. Zielgruppen

Das vorliegende Dokument wendet sich vornehmlich an Projektleiter, Softwarearchitekten und Softwareentwickler, die mit den Grundlagen des eXTra-Standards bereits vertraut sind, siehe insbesondere [KOMP].

Weitere Dokumente für die Zielgruppe der Projektleiter und Softwarearchitekten sind die Design Guidelines [DSIG], die Profilierung [PROF], die Versionierung [VERS], die Sicherheit und Verfügbarkeit [EXSEC] und gegebenenfalls eXTra und WebServices [EXWS].
Für die Zielgruppe der Softwarearchitekten und Softwareentwickler stehen darüber hinaus die Schnittstellenbeschreibungen für den eXTra-Basisstandard eXTra Transport [IFACE] und die der Standardnachrichten [EMSG] zur Verfügung,

2. Einführung

Zentrales Anliegen des eXTra-Standards ist, ihn so zu untergliedern, dass er zusammen mit der ⓘ Profilierung Gestaltungs- und Wahlmöglichkeiten eröffnet, die sowohl eine bedarfsorientierte Maßschneiderung als auch eine Anreicherung des zu konstruierenden ⓘ verbundspezifischen eXTra-Standards erlaubt.

Deshalb wurde der eXTra-Standard untergliedert in

- den Teil eXTra Transport, der den inneren Kern darstellt und sowohl Pflicht- als auch optionale Bestandteile enthält, z.B. die ⓘ PlugIns, und
- die optionalen eXTra-Standardnachrichten, die Sprachmittel für weitverbreitete Prozessabläufe zur Verfügung stellen.

Jeder dieser Bestandteile verfügt über entsprechende Schemadateien, die jeweils getrennt profiliert werden können. Bei der Gestaltung eines verbundspezifischen eXTra-Standards muss eXTra Transport eingesetzt werden, der je nach Bedarf profiliert werden kann, wobei die verpflichtenden XML-Tags in eXTra Transport nicht profiliert werden dürfen. Aus der Menge der PlugIns und der Standardnachrichten kann man dagegen je nach Bedarf beliebige Elemente auswählen und gegebenenfalls profilieren. Dabei ist jede Einschränkung einer Standardnachricht oder eines PlugIn in der Profilierung gestattet, Erweiterungen sind nicht zulässig. Zu den Regeln und Restriktionen bei der Profilierung siehe [PROF].

Alle eXTra-Standardnachrichten sind im Sinne der Profilierung und aus Sicht von eXTra Transport optional; hier gibt es keine Pflichtbestandteile. Sie sind im Sinne von eXTra fachliche Nachrichten, die der eXTra-Sender mit dem eXTra-Empfänger und entweder mit dessen eXTra Instanz des Distribution-Servers (derjenigen Instanz, die die gesendeten Daten an das gewünschte ⓘ Fachverfahren weiterreicht) oder mit dessen eXTra-Instanz des Delivery-Servers (derjenigen Instanz, die das Ausliefern angeforderter Daten organisiert) austauscht.

Bei der Analyse bestehender und neuer ⓘ Datenübermittlungsverfahren und deren Anforderungen haben sich insbesondere bei Fachverfahren, die die fachlichen Daten asynchron verarbeiten, unterschiedliche Prozessabläufe herauskristallisiert. Diese Prozessabläufe haben Themenschwerpunkte, die mit eXTra-Standardnachrichten unterstützt werden können. Diese Themenschwerpunkte sind:

- Unterstützung von Prozessketten, die aus mehreren Prozessen auf Sender- wie Empfängerseite bestehen können. Dies gilt insbesondere für Prozessketten, in die ein asynchron verarbeitendes Fachverfahren auf Empfängerseite involviert ist und bei denen der Sender sehr an den Verarbeitungsergebnissen seiner Sendungen interessiert ist.

  Der Sender kann die bereitgestellten (Ergebnis-) Daten mit der Standardnachricht DataRequest V1.2 anfordern und anschließend den erfolgreichen Abholprozess mit der Standardnachricht ConfirmationOfReceipt V1.3 bestätigen (dafür genügt die Schemadatei V1.3; siehe Kapitel 3.1).

  Ab der Schemadatei V1.4 der Standardnachrichten kann sich der Sender nicht nur – wie auf Basis der Schemadatei V1.3 - darüber informieren, ob der Empfänger seine Sendungen bereits verarbeitet hat und Verarbeitungsergebnisse erzeugt hat bzw. ob für ihn Daten bereitgestellt wurden, sondern auch über alle beim Empfänger erhältlichen (Ergebnis-) Daten, abhängig von deren Abholstatus (AVAILABLE, FETCHED, CONFIRMED). Zu diesem Zweck kann der Sender ein Inhaltsverzeichnis der erhältlichen (Ergebnis-) Daten des gewünschten Abholstatus mit der Standardnachricht ListRequest anfordern. Die Antwort erhält er in Form der Standardnachricht ListResponse. Mit diesen Informationen kann der Sender sich anschließend mit der erweiterten Standardnachricht DataRequest V1.3 gezielt die (Ergebnis-) Daten mit dem gewünschten Abholstatus abholen (siehe Kapitel 3.2).

- Unterstützung des automatischen, bedienerlosen Betriebs auf Sender- wie Empfängerseite.

  Dafür dient die Standardnachricht RepeatRequest (gleichermaßen für die Schemadatei V1.3 wie V1.4 der Standardnachrichten; siehe Kapitel 4)
- Unterstützung von Auskünften über den Bearbeitungsstatus von Sendungen und des Nachvollzugs über den Werdegang von Vorgängen auf Empfängerseite. Derartige Auskünfte kann der Sender mit der Standardnachricht StatusRequest anfordern und erhält die Antwort in Form der Standardnachricht StatusResponse (gleichermaßen für die Schemadatei V1.3 wie V1.4 der Standardnachrichten; siehe Kapitel 5).

In diesem Dokument werden die zugrunde liegenden Szenarien beschrieben, die zu den einzelnen eXTra-Standardnachrichten geführt haben, zudem wird ein Überblick über deren Ausgestaltung gegeben sowie Beispiele aufgezeigt.

3. eXTra-Standardnachrichten zur Unterstützung von Prozessketten

3.1. Einfache Szenarien zur Unterstützung von Prozessketten

Hinweis:
Die folgenden Ausführungen gelten gleichermaßen für die Schemadatei der Standardnachrichten der Version V1.3 wie auch V1.4.

Es gibt zwei einfache Szenarien, in die typischerweise die jeweiligen Prozessketten zwischen Sender und Empfänger eingeordnet werden können. Mit der Bezeichnung „einfaches Szenario“ soll einerseits zum Ausdruck gebracht werden, dass dieses Szenario bereits zu Beginn der Überlegungen zum eXTra-Standard betrachtet wurde. Andererseits ist mit „einfach“ gemeint, dass insbesondere der Erfolgsfall, also keine Sonder- und Fehlerfälle Eingang in die Betrachtung fanden.
In diese beiden Szenarien sind jeweils asynchron verarbeitende Fachverfahren involviert.
Realisiert das Fachverfahren auf Empfängerseite hingegen ein Dialogverfahren, verarbeitet die empfangenen Daten also sofort und teilt das Verarbeitungsergebnis dem Sender in der eXTra-Response sofort mit, so ist hierfür nach heutigem Erkenntnisstand keine spezifische Prozesskette bekannt, die mit einer eXTra-Standardnachricht unterstützt werden sollte.

Im Folgenden werden also nur Prozessketten behandelt, in die ein asynchron verarbeitendes Fachverfahren involviert ist.

Beim ersten Szenario besteht die Prozesskette aus eXTra-Sicht in der Regel aus drei Schritten, die jeweils der Sender initiiert:

(1) Dem Senden von fachlichen Meldungen, die der eXTra-Distribution-Server entgegennimmt und an das gewünschte Fachverfahren weiterreicht. Wann das spezifizierte Fachverfahren die fachlichen Meldungen verarbeitet, ist für den Sender unbekannt. Dessen ungeachtet ist der Sender sehr am Ergebnis der Verarbeitung

seiner fachlichen Daten durch das Fachverfahren auf Empfängerseite interessiert, selbst wenn dies erst nach geraumer Zeit zur Verfügung stehen sollte.

(2) Dem Anfordern von Protokollen, Ergebnissen oder resultierenden Daten der Verarbeitung durch das Fachverfahren auf Empfängerseite, allgemein dem Anfordern von fachlichen Nachrichten oder Daten. Die Anforderung wird durch den eXTra-Delivery-Server bearbeitet.

(3) Dem Bestätigen der erfolgreich abgeholten fachlichen Nachrichten oder Daten, die sich wiederum an den eXTra-Delivery-Server richtet.

Beim zweiten Szenario beginnt die Prozesskette aus Sicht von eXTra mit dem Anfordern von fachlichen Nachrichten oder Daten durch einen Sender, welche die andere Seite – der Empfänger – ausliefert. Für den Sender ist dabei nicht sichtbar, ob das Fachverfahren auf Empfängerseite die Nachrichten oder Daten bereits im Vorfeld dem eXTra-Delivery-Server zur Verfügung gestellt hat oder erst auf Anforderung durch den eXTra-Delivery-Server zur Auslieferung „on-the-fly“ bereitstellt. Der Datenfluss geht hier also im Vergleich zum ersten Szenario in umgekehrter Richtung. Diese Prozesskette besteht aus der Sicht von eXTra in der Regel aus zwei Schritten:

(1) Dem Anfordern von fachlichen Nachrichten bzw. Daten, die der Anforderer holen und verarbeiten will, z.B. von Rechnungen, oder von Stammdaten, die er übernehmen will.

(2) Dem Bestätigen der erfolgreich abgeholten fachlichen Nachrichten bzw. Daten durch den Anforderer.

In beiden Szenarien fordert ein Sender vom Empfänger fachliche Nachrichten oder Daten an und bestätigt dem Empfänger anschließend die erfolgreiche Abholung. Hier bot sich an, die Sprachmittel für die Anforderung von fachlichen Nachrichten bzw. Daten und der Bestätigung der erfolgreichen Abholung zu standardisieren und als eXTra-Standardnachrichten zur Verfügung zu stellen.

Die Motivation für den abschließenden Prozessschritt der Bestätigung erfolgreich abgeholter fachlicher Nachrichten bzw. Daten ist für die Empfängerseite der Zugewinn an Klarheit und Rechtssicherheit sowie an Effizienz in der Datenhaltung. Wenn die Empfängerseite z.B. gesetzlich verpflichtet ist, auf jeden Fall fachliche Nachrichten (z.B. Rückmeldungen) zur Verfügung zu stellen, dann ist mit der Bestätigung der erfolgreichen Abholung der Nachweis erbracht, dass die Empfängerseite ihre Verpflichtung erfüllt hat. Neben der Rechtsicherheit hat

die Empfängerseite mit der Bestätigung der erfolgreichen Abholung zugleich eine sichere Grundlage für eine anschließende Bereinigung der Datenhaltung des eXTra-Delivery-Servers.

3.1.1. Die Standardnachricht „DataRequest“

3.1.1.1. Gestaltung der Standardnachricht „DataRequest“

Die Standardnachricht **DataRequest** stellt für den Sender für die beiden benannten ursprünglichen Szenarien geeignete Sprachmittel zur Verfügung, die es ihm erlauben,

- fachliche Nachrichten, z.B. Rückmeldungen zu Lieferungen der jüngsten Vergangenheit, anzufordern (Szenario 1),

oder

- fachliche Nachrichten oder Daten anzufordern, die der Empfänger in jüngster Vergangenheit bereitgestellt hat oder bereitstellen soll (Szenario 2)

Die Sprachmittel von **DataRequest** müssen dabei so gestaltet sein, dass sie für alle möglichen Typen von Teilnehmern geeignet sind, also sowohl für einzelne Unternehmen, die im direkten Kontakt mit dem Empfänger oder den Empfängern stehen, als auch für Service-Rechenzentren, die im Auftrag vieler Unternehmen handeln und somit einen Massenbetrieb abdecken müssen. Damit geht die Forderung einher, dass man mit einem **DataRequest**-Aufruf die gewünschten fachlichen Nachrichten bzw. die bereitgestellten Daten möglichst genau spezifizieren kann, aber auch, dass man mit einem Aufruf viele fachlichen Nachrichten bzw. die bereitgestellten Daten anfordern und erhalten kann.
Diese beiden Anforderungen werden mit dem Element **Query** und dem Kindelement **Argument** bzw. einer Sequenz von Kindelementen **Argument** erfüllt.

Da man mit der **Query** der Standardnachricht **DataRequest** eine Menge von fachlichen Nachrichten bzw. Daten anfordern kann (z.B. mit <Argument property=“RequestID“> und <GT>), ist es insbesondere bei Service-Rechenzentren sinnvoll, die Menge an fachlichen Nachrichten bzw. Daten begrenzen zu können. Für die Begrenzung gibt es das Element **Control**.

Damit ergibt sich folgende Grobstruktur

<DataRequest version="1.x">
    <Query> ……</Query>
    <Control> …..</Control>
</DataRequest>

Das Element Query

Das Element **Query** kann eine Folge von n Kindelementen **Argument** enthalten. **Argument** selbst kennt drei Attribute: @event, @property und @type.

Argument/@property gibt die abzufragende Eigenschaft an.

Enthält die **Query** eine Folge von Kindelementen **Argument,** so werden für die Selektion die einzelnen Kindelemente **Argument** und dessen Attribut @property mit einem logischen UND verknüpft.

Das Element Argument

Das Element **Argument** verfügt über drei Attribute (@event, @property und @type) und hat als alternative Kindelemente EQ, GE, LE, GT, LT bzw. IN.
Argument/@property gibt die abzufragende Eigenschaft an. Das Kindelement EQ, GT usw. spezifiziert Vergleichsoperator und Vergleichswert, wohingegen mit dem Kindelement IN eine Trefferliste angegeben werden kann.
Für die beiden Attribute @event und @property des Elementes **Argument** gibt es die Codelisten EventNames und DataRequestPropertyNames, die die zulässigen Werte für @event bzw. @property festlegen.

Codeliste DataRequestPropertyNames

	xmsg:DataRequestPropertyNamesType
Inhalt	Identifikatoren abfragbare Eigenschaften
Datentyp	xsd:anyURI
Individualisierbar	Nein
Beliebige Domain	Nein

Vordefinierte Werte:

```
http://www.extra-standard.de/property/SenderID
http://www.extra-standard.de/property/ReceiverID
http://www.extra-standard.de/property/Procedure
http://www.extra-standard.de/property/DataType
http://www.extra-standard.de/property/ResponseID
http://www.extra-standard.de/property/ResponseCreationTimeStamp
http://www.extra-standard.de/property/ResponseFileName
http://www.extra-standard.de/property/Layer
```

Mit dem Attribut @event kann der Sender eindeutig formulieren, auf welchen Vorgang er sich bezieht, den er mit @property="ResponseID" nennt. Wenn der Sendevorgang der ursprünglichen Lieferung mittels eXTra-Request gemeint ist, dann muss er dies mit @event="SendData" bezeichnen. Wenn der letzte Holvorgang gemeint ist, dann muss er @event="RequestData" setzen.

Codeliste EventNames

	xmsg:EventNamesType
Inhalt	Identifikatoren von Ereignissen, die im Ablauf eines eXTra Kommunikationsszenarios auftreten können
Datentyp	xsd:anyURI
Individualisierbar	Nein
Beliebige Domain	Ja

Vordefinierte Werte:

```
http://www.extra-standard.de/event/SendData
http://www.extra-standard.de/event/RequestData
```

Hinweis: Voraussetzung für die Nutzung der ResponseID/ResponseFilename für die Selektion einer Menge von fachlichen Nachrichten bzw. Daten mittels Vergleichsoperator GT, GE, LT, LE ist, dass der eXTra-Server auf Empfängerseite die Begriffe zumindest pro Empfänger (ReceiverID) streng aufsteigend vergibt.

Hinweis: Kann der Sender die von ihm gewünschten fachlichen Nachrichten bzw. Daten exakt bezeichnen, so steht ihm hierfür das Kindelement IN zur Verfügung, mit der er eine Trefferliste angeben kann.

```
<xmsg:Query>

    <xmsg:Argument property="http://www.extra-standard.de/property/ResponseID" type="xs:string"
    event="http://www.extra-standard.de/event/RequestData">
            <xmsg:IN>
                    <xmsg:EQ>11112222</xmsg:EQ>
                    <xmsg:EQ>11112233</xmsg:EQ>
                    <xmsg:EQ>11112244</xmsg:EQ>
            </xmsg:IN>
    </xmsg:Argument>

</xmsg:Query>
```

Hinweis: Fehlt in der Query die Angabe für ReceiverID/Procedure/Datatype/ResponseID/ ResponseFileName, so wird dies jeweils als „keine Einschränkung“ interpretiert. Im Extremfall ist die Query leer oder besteht nur aus einem einzigen Argument, z.B.:

```
<xmsg:Query>

    <xmsg:Argument property="http://www.extra-standard.de/property/SenderID" type="xs:string">
            <xmsg:EQ>1111</xmsg:EQ>
    </xmsg:Argument>

</xmsg:Query>
```

In diesem Fall erhält der Sender von allen mit ihm in Beziehung stehenden Empfängern (alle vom Delivery-Server bedienten ReceiverIDs) alle für ihn bereitgestellten (und noch nicht als abgeholt bestätigten) fachlichen Nachrichten bzw. Daten. Umgangssprachlich formuliert sagt der Sender dem eXTra-Server auf Empfängerseite: „Gib mir alles, was du hast“.

Hinweis: Wie viele fachliche Nachrichten bzw. Daten der Sender in der zugehörigen eXTra-Response erhält, ist für ihn prinzipiell nicht vorhersehbar. Er kann jedoch die maximale Anzahl auszuliefernder fachlicher Nachrichten oder Pakete bzw. die maximale Datengröße mit dem Element **Control** festlegen.

Hinweis: Ein DataRequest mir leerer Query, also der Anforderung „Gib mir alles, was du hast“, könnte auf der Empfängerseite aufwändig zu bearbeiten sein. Insbesondere dann, wenn

überwiegend solche Anforderungen mit „keine Daten vorhanden“ beantwortet werden, hat dies eine unerwünschte Systembelastung auf Empfängerseite zur Folge. Die beiden Standardnachrichten ListRequest und ListResponse könnten hier Entlastung bieten (siehe 3.2).

Das Element Control

Mit dem Element **Control** kann man die Menge an fachlichen Nachrichten bzw. bereitgestellten Daten auf ein definiertes Maß beschränken und so einen geordneten Betrieb sicherstellen. Dies ist für Service-Rechenzentren und ihrem Massenbetrieb wichtig, insbesondere wenn z.B. nach länger andauernden Störungen der Betrieb wieder aufgenommen wird und inzwischen eine große Menge an bereitstehenden fachlichen Nachrichten oder Daten aufgelaufen ist.

Die Beschränkung kann alternativ mit dem Kindelement **MaximumPackages, MaximumMessages** oder **MaximumSize** formuliert werden.

Hinweis zur anschließenden Auslieferung von fachlichen Nachrichten bzw. Daten, d.h. zur eXTra-Response auf einen vorangegangenen eXTra-Request mittels **DataRequest:**
Um eine klare Zuordnung von fachlichen Nachrichten/Rückmeldungen zu vorangegangenen Lieferungen zu ermöglichen, ist es erforderlich, dass das verarbeitende Fachverfahren auf Empfängerseite zu jeder gesendeten Message/Package/Lieferung eine dazugehörige Rückmeldung erzeugt. Die Rückmeldung sollte als eXTra-Package oder Message bereitgestellt werden, weil damit in der eXTra-Response auf einen DataRequest mehrere Rückmeldungen in Form von Messages oder Packages ausgeliefert werden können.

3.1.1.2. Beispiel für die Standardnachricht „DataRequest“ der Version V1.2

Es werden die fachlichen Nachrichten/Rückmeldungen zu allen Lieferungen/Packages/Messages angefordert, die der angegebenen Selektion entsprechen. Die Selektion des Beispiels entspricht typischerweise der eines Service-Rechenzentrums, das alle inzwischen aufgelaufenen fachlichen Nachrichten bzw. Daten eines Empfängers für genau einen Empfänger und ein Fachverfahren anfordert.

```
<xmsg:DataRequest version=”1.2”>

    <xmsg:Query>

        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>12347259</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:EQ>Meldung</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseID"
          event=”RequestData” type="xs:string">
                <xmsg:GT>009999</xmsg:GT>
        </xmsg:Argument>

    </xmsg:Query>


    <xmsg:Control>
          <xmsg:MaximumPackages>100</xmsg:MaximumPackages>
          <!--    Begrenzung der Paketanzahl auf 100 … →
    </xmsg:Control>

</xmsg:DataRequest>
```

Die angegebene ResponseID=009999 ist – wegen event=“RequestData“ - die ResponseID des eXTra-Response auf den zuletzt gesendeten DataRequest.

Alternative zu ResponseID ist ResponseFilename.

3.1.2. Die Standardnachricht „ConfirmationOfReceipt“

3.1.2.1. Gestaltung der Standardnachricht „ConfirmationOfReceipt“

Nachdem beim Szenario 1 im Prozessschritt (2) der Sender fachliche Nachrichten oder Daten erfolgreich abgeholt hat, kann er dies dem Empfänger im Prozessschritt (3) mit der eXTra-Standardnachricht **ConfirmationOfReceipt** auch explizit für jede fachliche Nachricht oder Datei mit einer Folge von Kriterien – jeweils formuliert mit dem Attribut @name im Element **Property** - bestätigen.

Analoges gilt für das Szenario 2 und den beiden Prozessschritten Abholen (1) und Bestätigen (2).

Die Wirkung der Standardnachricht **ConfirmationOfReceipt** ist, dass der eXTra-Deliveryserver auf Empfängerseite die bestätigten fachlichen Nachrichten bzw. Daten als abgeholt markiert und bei einem anschließenden eXTra-Request mit **DataRequest** nicht mehr ausliefert.

Die mit Property/@name formulierbaren Kriterien sind in folgender Codeliste aufgeführt, der Wert des jeweiligen Kriteriums wird mit dem Kindelement **Value** angegeben:

Codeliste PropertyNames

	xmsg: PropertyNamesType
Inhalt	Menge allgemein abfragbarer Eigenschaften
Datentyp	xsd:anyURI
Individualisierbar	Nein
Beliebige Domain	Ja

Vordefinierte Werte:

```
http://www.extra-standard.de/property/SenderID
http://www.extra-standard.de/property/ReceiverID
http://www.extra-standard.de/property/Procedure
http://www.extra-standard.de/property/DataType
http://www.extra-standard.de/property/RequestID
http://www.extra-standard.de/property/RequestCreationTimeStamp
http://www.extra-standard.de/property/RequestFileName
http://www.extra-standard.de/property/ResponseID
http://www.extra-standard.de/property/ResponseCreationTimeStamp
http://www.extra-standard.de/property/ResponseFileName
http://www.extra-standard.de/property/Layer
```

Sollen mehrere fachliche Nachrichten oder Dateien, auf die ein gleiches Kriterium zutrifft, z.B. ResponseID oder ResponseFileName, mit einer einzigen eXTra-Standardnachricht **ConfirmationOfReceipt** bestätigt werden, so steht hierfür das Element **PropertySet** zur Verfügung.
Werden mehrere Kriterien in einer Folge von Elementen **Property** und evtl. abschließenden **PropertySet** angegeben, so werden – analog zur Standardnachricht **DataRequest** - die einzelnen Elemente der Folge mit einem logischen UND verknüpft.

Wird als Kriterium @name="ResponseFileName" verwendet, so müssen die empfangenen fachlichen Nachrichten oder Dateien einzeln bestätigt werden.

Nutzt man hingegen als Kriterium z.B. @name="ResponseID", so ist damit nicht eindeutig festgelegt, ob damit die ResponseID der Transport-Ebene oder der Paket- bzw. Message-Ebene gemeint ist. Ursache hierfür ist die Auslieferung von fachlichen Nachrichten bzw. Daten auf Grund eines DataRequest in Form von eXTra-Packages oder eXTra-Messages (siehe oben Hinweis zu DataRequest). Mit dem Kriterium @layer kann die erforderliche Eindeutigkeit erzielt werden.
Mit der Angabe @layer="Transport" @name="ResponseID" bestätigt der Sender alle fachlichen Nachrichten bzw. Daten, die er in der eXTra-Response eines DataRequest mit dieser ResponseID im TransportHeader erhalten hat.

3.1.2.2. Beispiel für die Standardnachricht „ConfirmationOfReceipt"

Beispiel 1 für eine Bestätigung mit dem Kriterium @name="ResponseFileName", wie sie für die Gesetzlichen Krankenkassen und deren GKV-Kommunikationsserver typisch ist:

```
<xmsg:ConfirmationOfReceipt version="1.3">

        <xmsg:Property name="http://www.extra-standard.de/property/SenderID"
                type="xs:string">
                <xmsg:Value>12341111</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID"
                type="xs:string">
                <xmsg:Value>12348888</xmsg:Value>
        </xmsg:Property>


        <xmsg:PropertySet name="http://www.extra-standard.de/property/ResponseFilename"
                type="xs:string">
        <!-- Bestaetigung von 3 Rueckmeldungen jeweils mit dem entsprechenden Dateinamen
        -->
                <xmsg:Value>EDUA0007261</xmsg:Value>
                <xmsg:Value>EDUA0007262</xmsg:Value>
                <xmsg:value>EDUA0007266</xmsg:value>

         </xmsg:PropertySet>

</xmsg:ConfirmationOfReceipt>
```

Beispiel 2 für eine Bestätigung mit dem Kriterium @name="ResponseID", wie sie für die Rentenversicherung bei Sofortmeldungen typisch ist (Darstellung in Kurzform ohne SenderID und ReceiverID):

```
<xmsg:ConfirmationOfReceipt version="1.3">

        <xmsg:PropertySet name="http://www.extra-standard.de/property/ResponseID" layer="Package"
                type="xs:string">
        <!-- Bestaetigung von 3 Rueckmeldungen jeweils mit der entsprechenden ResponseID
        -->
                <xmsg:Value>7261</xmsg:Value>
                <xmsg:Value>7262</xmsg:Value>
                <xmsg:value>7266</xmsg:value>

         </xmsg:PropertySet>

 </xmsg:ConfirmationOfReceipt>
```

Beispiel3 für eine Sammelbestätigung mit dem Kriterium @name="ResponseID" und @layer="Transport" (Darstellung in Kurzform ohne SenderID und ReceiverID):

```
<xmsg:ConfirmationOfReceipt version="1.3">

        <xmsg:Property name="http://www.extra-standard.de/property/ResponseID" layer="Transport"
                type="xs:string">
        <!-- Bestaetigung von allen abgeholten Rueckmeldungen mit der entsprechenden ResponseID
        des TransportHeaders -->
                <xmsg:Value>7200</xmsg:Value>

         </xmsg:Property>

 </xmsg:ConfirmationOfReceipt>
```

3.2. Erweiterte Szenarien zur Unterstützung von Prozessketten (Schemadatei V1.4)

Die inzwischen erworbenen Erfahrungen mit dem eXTra-Standard im laufenden Betrieb haben für manche Szenarien den Wunsch nach einer verbesserten Unterstützung laut werden lassen, die zur neuen Schemadatei der Version V1.4 der eXTra-Standardnachrichten geführt haben.

Beim Holprozess gibt die Unwägbarkeit des Bearbeitungsaufwands eines DataRequest auf Empfängerseite bzw. der nicht vorhersehbare Umfang der zugehörigen eXTra-Response - trotz der Möglichkeit der Beschränkung mit dem Element Control - Anlass zur Kritik (siehe auch die Beispiele zu ListOfDataRequest und ListOfConfirmationOfReceipt). Hieraus erwächst der Wunsch, fachliche Nachrichten bzw. Daten gezielt anfordern zu können.

Bei einem weiteren Szenario des Holprozesses wäre ebenfalls eine bessere Unterstützung durch den eXTra-Standard wünschenswert: Erleidet der Sender einen Datenverlust, z.B. durch einen gravierenden Hardwarefehler, wobei seit der letzten Datensicherung einige Tage vergangen sind, so fehlen ihm zunächst alle Informationen über den zuletzt stattgefundenen Datenaustausch in beide Richtungen. Er hat dadurch alle Informationen verloren, die ihm sagen, welche fachlichen Daten er nach der letzten Datensicherung gesendet hat, zudem weiß er nicht welche fachlichen Nachrichten bzw. Daten er seitdem abgeholt und bestätigt hat. Mit der Standardnachricht StatusRequest (siehe 5.2) kann er sich bezüglich seines Sendebetriebs über den Zustand auf Empfängerseite informieren und so über die Standardnachricht StatusResponse (siehe 5.3) in Erfahrung bringen, welche Sendevorgänge mit welchem Ergebnis seit der letzten Datensicherung erfolgt sind. Im Vollausbau auf Empfängerseite kann er mittels StatusResponse auch erfahren, welche Holvorgänge er seitdem angestoßen und bereits bestätigt hat. Zusätzlich bräuchte er aber noch die Möglichkeit, die von ihm bereits abgeholten und noch nicht bestätigten sowie die bereits bestätigten fachlichen Nachrichten oder Daten **nochmals abzuholen**, um den Systemzustand auf seiner Senderseite wieder herzustellen, der vor dem Datenverlust bestand.

Die Möglichkeit, bereits abgeholte und bestätigte fachliche Nachrichten bzw. Daten erneut abholen zu können, ergibt sich ebenfalls bei folgendem Szenario als Konsequenz aus der Forderung nach einer effizienten Datenhaltung auf Empfängerseite, die möglichst unabhängig von der Senderseite sein soll. Derzeit verhält sich eine nennenswerte Anzahl von Sendern beim

Abhol- und Bestätigungsprozess für die Empfängerseite nicht wunschgemäß. Diese Sender holen zwar ihre fachlichen Nachrichten bzw. Daten zumeist regelmäßig ab, allerdings bestätigen sie den Erhalt der fachlichen Nachrichten bzw. Daten erst sehr spät, oder sie vergessen sogar die Bestätigung gänzlich, was den Datenbestand auf Empfängerseite unnötig vergrößert. Im Falle der gesetzlichen Krankenkassen und der Rentenversicherung müssen dann die Sender gezielt und explizit zum Abholen und Bestätigen ihrer Daten aufgefordert werden, bei Ablauf der vorgegebenen Zustellfristen müssen die Meldungen auf kostenintensiven Ersatzwegen (z.B. Briefpost) zugestellt werden. Deshalb wäre es für die Empfängerseite ideal, wenn die Sender den Abhol- und Bestätigungsprozess zu einem integrierten Vorgang zusammenfassen würden. Für die Sender ist dies aber nur dann akzeptabel, wenn sie Unterstützung für den Fall bekommen, dass die von ihnen gerade abgeholten und bestätigten fachlichen Nachrichten bzw. Daten aus irgendwelchen Gründen nicht bei ihren Fachverfahren angekommen sind und sie vor der Notwendigkeit stehen, den integrierten Abhol- und Bestätigungsprozess wiederholen zu können.

Zusammenfassend bieten die eXTra-Standardnachrichten auf Basis der Schemadatei V1.3 keine ausreichenden Sprachmittel, um sich mit einer Auskunftsfunktion einen Überblick über den aktuellen Betrieb und den Gesamtsystemzustand zu verschaffen. Eine gezielte Anforderung spezifizierter fachlicher Nachrichten bzw. Daten ist damit nicht möglich, der Aufwand für den Empfänger und das Ergebnis für den Sender ist nicht vorhersehbar. Eine Wiederholung des Abhol- und Bestätigungsprozesses ist ebenfalls nicht möglich.

Abhilfe schaffen zwei weitere Standardnachrichten in Kombination mit der erweiterten Standardnachricht **DataRequest**, die mit der neuen Schemadatei V1.4.0 zur Verfügung gestellt werden.
Mit der Standardnachricht **ListRequest** (siehe 3.2.1) kann der Sender Auskunft über den aktuellen Zustand auf Empfängerseite anfordern und mit der Antwort in Form der Standardnachricht **ListResponse** (siehe 3.2.2) das Ergebnis der Anfrage erhalten, sozusagen ein Inhaltsverzeichnis oder eine Trefferliste. Neben der reinen Auskunftsfunktion bieten diese beiden Standardnachrichten die Grundlage für eine erweiterte Standardnachricht **DataRequest** (siehe 3.2.3), mit welcher der Sender einerseits gezielt bereitgestellte fachliche Nachrichten bzw. Daten anfordern kann und andererseits bei Bedarf Abhol- und Bestätigungsprozesse wiederholen kann.

3.2.1. Die Standardnachricht „ListRequest“

3.2.1.1. Gestaltung der Standardnachricht „ListRequest“

Genau wie die Standardnachricht **DataRequest** oder **StatusRequest** verfügt **ListRequest** über eine Query, mit deren Hilfe die gewünschte Auskunft formuliert werden kann. Um den Umfang der Rückmeldung per **ListResponse** begrenzen zu können, gibt es das Element **Control.**

Das Ergebnis eines **ListRequest** ist ein Inhaltsverzeichnis, das die Empfängerseite in Form der Standardnachricht **ListResponse** ausliefert (siehe 3.2.2).

Für die Standardnachricht **ListRequest** ergibt sich folgende Grobstruktur:

```
<ListRequest version=“1.0“>
        <Query>
                <Argument property=”…“>…..</Argument>
                ….
                <Argument property=”…“>…..</Argument>
        </Query>
        <Control> …..</Control>

</ListRequest>
```

Das Element Query

Der formale Aufbau der **Query** ist analog zur Query der Standardnachricht **DataRequest** (siehe 3.1.1) bzw. **StatusRequest** (siehe 5.2), kann also aus einer Folge von n Kindelementen **Argument** bestehen.

Das Element Argument

Das Element **Argument** kennt wie bei der Standardnachricht **DataRequest** oder **StatusRequest** die drei bekannten Attribute @property, @event und @type. Zusätzlich gibt es

das Attribut @order, mit dem die Sortierreihenfolge innerhalb des angeforderten Inhaltsverzeichnisses festgelegt werden kann.
Die Verwendung und Bedeutung der Attribute @event und und @type ist identisch zu **DataRequest** (siehe 3.1.1) oder **StatusRequest** (siehe 5.2)..

Das Attribut @property gibt wie üblich eine abzufragende Eigenschaft an, die in der folgenden Codeliste ListRequestPropertyNames enthalten sein muss.
Die **ListRequest** zugeordnete Codeliste entspricht derjenigen von **StatusRequest** (siehe 5.2), die um die Eigenschaft State erweitert wurde. Damit ergibt sich für die Codeliste ListRequestPropertyNames:

Codeliste ListRequestPropertyNames

	xmsg:ListRequestPropertyNamesType
Inhalt	Identifikatoren abfragbare Eigenschaften
Datentyp	`xsd:anyURI`
Individualisierbar	Nein
Beliebige Domain	Nein

Vordefinierte Werte:

```
http://www.extra-standard.de/property/SenderID
http://www.extra-standard.de/property/ReceiverID
http://www.extra-standard.de/property/Procedure
http://www.extra-standard.de/property/DataType
http://www.extra-standard.de/property/RequestID
http://www.extra-standard.de/property/ResponseID
http://www.extra-standard.de/property/RequestCreationTimeStamp
http://www.extra-standard.de/property/ResponseCreationTimeStamp
http://www.extra-standard.de/property/RequestFileName
http://www.extra-standard.de/property/ResponseFileName
http://www.extra-standard.de/property/Layer
http://www.extra-standard.de/property/State
```

Das neue Property state kann folgende Werte annehmen:

```
http://www.extra-standard.de/requeststate/NONEhttp://www.extra-
standard.de/requeststate/AVAILABLE
http://www.extra-standard.de/requeststate/FETCHED
http://www.extra-standard.de/requeststate/CONFIRMED
```

Die Werte bedeuten:

NONE	keine Treffer zur gegebenen Query
AVAILABLE	bereitgestellt, aber noch nicht abgeholt
FETCHED	abgeholt aber noch nicht bestätigt
CONFIRMED	abgeholt und bereits bestätigt

Mit dem Attribut @order kann man die Sortierreihenfolge innerhalb des angeforderten Inhaltsverzeichnisses pro Eigenschaft (Property) festlegen.

Das Attribut @order kann folgende Werte annehmen:

A|ASC|ASCENDING	aufsteigend
D|DESC|DESCENDING	absteigend

Hinweis: wird das Attribut @order bei mehreren Eigenschaften (Property) angegeben, so wird entsprechend der Reihenfolge der Eigenschaften sortiert.

```
<xmsg:Query>

    <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" order=”ASC”>
      <xmsg:IN>
            <xmsg:EQ>11112222</xmsg:EQ>
            <xmsg:EQ>55556666</xmsg:EQ>
      </xmsg:IN>
    </xmsg:Argument>

    <xmsg:Argument property="http://www.extra-standard.de/property/State" order="ASC">
      <xmsg:EQ> http://www.extra-standard.de/property/requeststate/AVAILABLE </xmsg:EQ>
    </xmsg:Argument>

    <xmsg:Argument property=”http://www.extra-standard.de/property/ResponseID” order=”ASC”>
            <xmsg:GT>009999</xmsg:GT>
    </xmsg:Argument>

</xmsg:Query>
```

In diesem Beispiel wird das Inhaltsverzeichnis zuerst aufsteigend nach den beiden Empfängern, dann nur die fachlichen Nachrichten/ Dateien mit dem Status „noch nicht abgeholt“ und zuletzt aufsteigend nach ihrer ResponseID sortiert.

Das Element Control

Mit dem Element **Control** kann man den Umfang der Rückmeldung per **ListResponse** auf ein definiertes Maß beschränken und so einen geordneten Betrieb sicherstellen. **Control** wird analog zur Standardnachricht **StatusRequest** definiert (siehe 5.2).

Die Beschränkung kann alternativ mit dem Kindelement **MaximumSize** oder **MaximumResults** formuliert werden.

3.2.1.2. Beispiele für die Standardnachricht „ListRequest“

Der Sender fordert Auskunft über alle Holvorgänge an, die dem spezifizierten Status und der Selektion mittels Query entspricht.

Beispiel 1:

Die Selektion dieses Beispiels entspricht typischerweise der eines Service-Rechenzentrums, das Auskunft über alle seit dem letzten Abholvorgang verfügbaren aber noch nicht abgeholten fachlichen Nachrichten bzw. Daten genau eines Empfängers und eines Fachverfahrens anfordert, aufsteigend nach ResponseID sortiert.

Anhand der erhaltenen Auskunft will das Service-Rechenzentrum in einem Folgeschritt gezielt fachliche Nachrichten bzw. Daten mittels erweitertem DataRequest (siehe 3.2.3) anfordern.

```
<xmsg:ListRequest version=”1.0”>
    <xmsg:Query>
        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>12347259</xmsg:EQ>
        </xmsg:Argument>
        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>
        <xmsg:Argument property="http://www.extra-standard.de/property/State">
                <xmsg:EQ>http://www.extra-standard.de/property/requeststate/AVAILABLE</xmsg:EQ>
        </xmsg:Argument>
        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseID" order=”ASC”
          event=”RequestData” type="xs:string“>
                <xmsg:GT>009999</xmsg:GT>
        </xmsg:Argument>
```

```
    </xmsg:Query>

</xmsg:ListRequest>
```

Die angegebene ResponseID=009999 ist – wegen event="RequestData" - die ResponseID des eXTra-Response auf den zuletzt gesendeten DataRequest.

Alternative zu ResponseID ist ResponseFilename.

Hinweis: Die Verwendung der ResponseID bei der **ListRequest**-Nachricht setzt ein bestimmtes Procedere auf Empfängerseite zwischen Fachverfahren und eXTra-Delivery-Server voraus. Das Fachverfahren muss nach einem Verarbeitungslauf die zugeordneten fachlichen Nachrichten bzw. Daten entweder sofort an den eXTra-Delivery-Server weiterreichen, so dass dieser bereits zu diesem Zeitpunkt die ResponseIDs bilden kann, zusammen mit der Übernahme der fachlichen Nachrichten bzw. Daten in seine Datenhaltung. Oder der eXTra-Delivery Server fordert sowohl bei einem **ListRequest** als auch einem **DataRequest** des Senders die bereitstehenden fachlichen Nachrichten bzw. Daten vom Fachverfahren an, bildet die ResponseIDs und übernimmt gleichzeitig die fachlichen Nachrichten bzw. Daten in seine Datenhaltung.
Ist das Zusammenspiel auf Empfängerseite zwischen Fachverfahren und eXTra-Delivery-Server jedoch anders organisiert, indem z.B. die ResponseID immer erst anlässlich eines **DataRequest** im Zuge der Auslieferung der fachlichen Nachrichten bzw. Daten gebildet wird, dann verbleibt für den **ListRequest** statt der ResponseID nur der ResponseFileName.

Beispiel 2:

Für dieses Beispiel wird angenommen, dass den Sender das Unglück einer gravierenden Hardwarestörung und eines Datenverlustes seit der letzten Datensicherung getroffen hat , wobei dies schon eine Woche zurückliegt. Die Selektion dieses Beispiels entspricht typischerweise der eines Senders, der Auskunft über alle seit der letzten Datensicherung von ihm abgeholten und noch nicht oder bereits bestätigten, aber leider verloren gegangenen fachlichen Nachrichten bzw. Daten will, um diese in einem Folgeschritt nochmals mittels erweitertem DataRequest (siehe 3.2.3) gezielt abholen und an sein Fachverfahren weiterreichen zu können.

```
<xmsg:ListRequest version=”1.0”>

    <xmsg:Query>

        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>12347777</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property=”http://www.extra-standard.de/property/State” order=”ASC”>
          <xmsg:IN>
                <xmsg:EQ> http://www.extra-standard.de/property/requeststate/FETCHED </xmsg:EQ>
                <xmsg:EQ> http://www.extra-standard.de/property/requeststate/CONFIRMED
                        </xmsg:EQ>
          </xmsg:IN>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseFileName"
          event=”RequestData” order=”ASC”>
                <xmsg:GT>EDUA007777</xmsg:GT>
        </xmsg:Argument>

    </xmsg:Query>


</xmsg:ListRequest>
```

Der angegebene ResponseFileName=007777 ist – wegen event=“RequestData“ – der Dateiname der – laut der beim Sender vorliegenden Daten - zuletzt abgeholten und ggfls bestätigten Datei.

Das Inhaltsverzeichnis soll zuerst aufsteigend nach dem Status und innerhalb des Status aufsteigend nach Dateinamen sortiert werden.

3.2.2. Die Standardnachricht „ListResponse“

3.2.2.1. Gestaltung der Standardnachricht „ListResponse“

Prinzipiell könnte man es dem Empfänger überlassen, wie er auf die Anforderung eines **ListRequest** reagiert und welche Informationen er zurückmeldet. Im Sinne einer Standardisierung von Vorgängen und deren Bearbeitung ist jedoch der Ansatz vorzuziehen, auch für die Rückmeldung eine weitere eXTra-Standardnachricht – **ListResponse** – zur Verfügung zu stellen.

Der Empfänger meldet als Antwort auf einen **ListRequest** je nach spezifizierter **Query** und Abholstatus den Zustand der Empfängerseite bezüglich der in der Vergangenheit liegenden Abhol- und Bestätigungsvorgänge (FETCHED bzw. CONFIRMED) oder der noch ausstehenden Abholvorgänge (AVAILABLE) in Form eines Inhaltsverzeichnisses bzw. einer Trefferliste zurück.

Das Inhaltsverzeichnis der **ListResponse** enthält für jede fachliche Nachricht bzw. Datei, auf die die **Query** der **ListRequest** zutrifft, jeweils ein Element **Entry**, das die spezifischen Eigenschaften der jeweiligen fachlichen Nachricht bzw. Datei auflistet.

Damit ergibt sich folgende Grobstruktur für die Standardnachricht **ListResponse**

```
<ListResponse version=“1.0“>
        <Entry> ……</Entry>
……
        <Entry> ……</Entry>
</ListResponse>
```

Das Element **Entry** listet für genau eine fachliche Nachricht bzw. Datei deren Eigenschaften in einer Folge von **Property** Elementen auf:

<u>Das Element Property</u>

Das Element **Property** ist von der Standardnachricht **ConfirmationOfReceipt** bzw. **StatusResponse** her bereits bekannt und hat den gleichen formalen Aufbau; es kann

mehrfach vorkommen, um die verschiedenen Eigenschaften einer fachlichen Nachricht bzw. Datei darzustellen.

Das Element **Property** hat wie bekannt drei Attribute: (@name, @type und @event) und kennt als Kindelement nur das Element **Value**.

Damit ergibt sich folgende bekannte Grobstruktur für das Element **Property:**

```
<Property name="…" type="…" event="…">
        <Value>…..</Value>
</Property>
…..
```

Die mit Property/@name formulierbaren Eigenschaften sind in folgender Codeliste aufgeführt, die – gegenüber der Codeliste StatusRequestPropertyNames (siehe 3.3.1) - um die Eigenschaft **PayloadSize** erweitert wurde. **PayloadSize** gibt die Größe einer fachlichen Nachricht bzw. Datei an, d.h. die Größe des Elementes **Data** im MessageBody oder PackageBody.

Codeliste ListResponsePropertyNames

	xmsg: ListResponsePropertyNamesType
Inhalt	Menge allgemein abfragbarer Eigenschaften
Datentyp	`xsd:anyURI`
Individualisierbar	Nein
Beliebige Domain	Ja

Vordefinierte Werte:

```
http://www.extra-standard.de/property/SenderID
http://www.extra-standard.de/property/ReceiverID
http://www.extra-standard.de/property/Procedure
http://www.extra-standard.de/property/DataType
http://www.extra-standard.de/property/RequestID
http://www.extra-standard.de/property/RequestCreationTimeStamp
http://www.extra-standard.de/property/RequestFileName
http://www.extra-standard.de/property/ResponseID
http://www.extra-standard.de/property/ResponseCreationTimeStamp
http://www.extra-standard.de/property/ResponseFileName
http://www.extra-standard.de/property/PayloadSize
http://www.extra-standard.de/property/State
```

Hinweis: Wenn die Empfängerseite die auf Grund eines **DataRequest** angeforderten fachlichen Nachrichten bzw. Daten in Form von Packages oder Messages ausliefert, dann entstammen die im **Entry** aufgelisteten Eigenschaften einem Package- bzw. Message-Header. Die Empfängerseite versucht, für jede fachliche Nachricht (bzw. Daten) das Maximum an verfügbaren Informationen zur Verfügung zu stellen.

Hinweis: Wenn der Sender das Inhaltsverzeichnis für fachlichen Nachrichten bzw. Daten anfordert, die er noch nicht abgeholt hat, also mit Status AVAILABLE, dann wird es beim Element **Entry** zumindest keine RequestID geben können, weil es noch keinen entsprechenden **DataRequest** Aufruf gab. Ob zu diesem Zeitpunkt bereits eine ResponseID existiert, ist vom Zusammenspiel zwischen Fachverfahren und eXTra-Delivery-Server auf Empfängerseite abhängig (siehe auch Hinweis zu **ListRequest** Beispiel1 unter 3.2.1.2)

3.2.2.2. Beispiele für die Standardnachricht „ListResponse“

Beispiel 1

Ausgangspunkt sei die ListRequest Anforderung unter 3.2.1.2 Beispiel 1:

Der Sender will ein Inhaltsverzeichnis der inzwischen erzeugten, aber noch nicht abgeholten fachlichen Nachrichten bzw. Daten des Fachverfahrens DEÜV seit dem letzten Abholvorgang mit der ResponseID=009999.

```
<xmsg:ListRequest version=”1.0”>
    <xmsg:Query>

        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>12347259</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/Property/state">
                <xmsg:EQ> http://www.extra-standard.de/property/requeststate/AVAILABLE </xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseID" order=”ASC”
          event=”RequestData” type="xs:string“>
                <xmsg:GT>009999</xmsg:GT>
        </xmsg:Argument>

    </xmsg:Query>
```

</xmsg:ListRequest>

Die Antwort in Form einer **ListResponse** könnte folgendermaßen aussehen:

```
<xmsg:ListResponse version="1.0">

        <xmsg:Entry>
                <xmsg:Property name="http://www.extra-standard.de/property/SenderID">
                        <xmsg:Value>11112222</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID">
                        <xmsg:Value>12347259</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/Procedure">
                        <xmsg:Value>DUA</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/DataType">
                        <xmsg:Value>Meldung</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/ResponseID"
                        event="RequestData" type="xs:string">
                        <xmsg:Value>0100005</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-
                        standard.de/property/ResponseCreationTimestamp">
                        <xmsg:Value>2013-01-17T07:39:50</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/ResponseFileName">
                        <xmsg:Value> EDUA001111</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/PayloadSize">
                        <xmsg:Value>1234567B</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/State">
                        <xmsg:Value>http://www.extra-
                                standard.de/property/requeststate/AVAILABLE</xmsg:Value>
                </xmsg:Property>
        </xmsg:Entry>

……….

        <xmsg:ListEntry>
        <xmsg:Entry>
                <xmsg:Property name="http://www.extra-standard.de/property/SenderID">
                        <xmsg:Value>11112222</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID">
                        <xmsg:Value>12347259</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name="http://www.extra-standard.de/property/Procedure">
                        <msg:Value>DUA</xmsg:Value>
                </xmsg:Property>
```

```
            <xmsg:Property name=”http://www.extra-standard.de/property/DataType“>
                  <xmsg:Value>Meldung</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/ResponseID“
                  event=”RequestData” type="xs:string“>
                  <xmsg:Value>0100012</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-
                  standard.de/property/ResponseCreationTimestamp“>
                  <xmsg:Value>2013-01-18T13:30:10</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/ResponseFileName“>
                  <xmsg:Value> EDUA001115</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/PayloadSize“>
                  <xmsg:Value>2222267B</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/State“>
                  <xmsg:Value>http://www.extra-
                        standard.de/property/requeststate/AVAILABLE</xmsg:Value>
            </xmsg:Property>
      </xmsg:Entry>

</xmsg:ListResponse>
```

Hinweis: Da es beim angefragten Abholstatus AVAILABLE noch keinen entsprechenden **DataRequest**-Aufruf geben konnte, fehlt beim Element **Entry** der Eintrag einer RequestID.

Beispiel 2

Ausgangspunkt sei die ListRequest Anforderung unter 3.2.1.2 Beispiel 2:

Der Sender will ein Inhaltsverzeichnis aller der von ihm bereits abgeholten und bestätigten fachlichen Nachrichten bzw. Daten des Fachverfahrens DEÜV, die der Empfänger nach der Datei mit dem Namen EDUA007777 in aufsteigender Reihenfolge erzeugt hat.

```
<xmsg:ListRequest version=”1.0”>

   <xmsg:Query>

      <xmsg:Argument property="http://www.extra-standard.de/Property/state">
        <xmsg:IN>
            <xmsg:EQ> http://www.extra-standard.de/property/requeststate/FETCHED </xmsg:EQ>
            <xmsg:EQ> http://www.extra-standard.de/property/requeststate/CONFIRMED
                  </xmsg:EQ>
        </xmsg:IN>
      </xmsg:Argument>

      <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
            <xmsg:EQ>12347259</xmsg:EQ>
```

```
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:EQ>Meldung</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseFileName"
          event=”RequestData” type="xs:string" order=”ASC”>
                <xmsg:GT>EDUA007777</xmsg:GT>
        </xmsg:Argument>

    </xmsg:Query>

</xmsg:ListRequest>
```

Die Antwort in Form einer **ListResponse** könnte folgendermaßen aussehen:

```
<xmsg:ListResponse version=“1.0“>

        <xmsg:Entry>
                <xmsg:Property name=”http://www.extra-standard.de/property/SenderID“>
                        <xmsg:Value>11112222</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ReceiverID“>
                        <xmsg:Value>12347259</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/Procedure“>
                        <xmsg:Value>DUA</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/DataType“>
                        <xmsg:Value>Meldung</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/RequestD“
                        event=”RequestData” type="xs:string“>
                        <xmsg:Value>20130103-01111</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ResponseID“
                        event=”RequestData” type="xs:string“>
                        <xmsg:Value>0008111</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-
                        standard.de/property/ResponseCreationTimestamp“>
                        <xmsg:Value>2013-01-02T09:09:11</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ResponseFileName“>
                        <xmsg:Value> EDUA007778</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/PayloadSize“>
```

```
                <xmsg:Value>12345B</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/State“>
                <xmsg:Value>http://www.extra-
                    standard.de/property/requeststate/CONFIRMED</xmsg:Value>
            </xmsg:Property>
        </xmsg:Entry>


        <xmsg:Entry>
            <xmsg:Property name=”http://www.extra-standard.de/property/SenderID“>
                <xmsg:Value>11112222</msg:Value>
            </msg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/ReceiverID“>
                <msg:Value>12347259</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/Procedure“>
                <xmsg:Value>DUA</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/DataType“>
                <xmsg:Value>Meldung</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/RequestD“
                event=”RequestData” type="xs:string“>
                <xmsg:Value>20130108-03333</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/ResponseID“
                event=”RequestData” type="xs:string“>
                <xmsg:Value>0010222</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-
                standard.de/property/ResponseCreationTimestamp“>
                <xmsg:Value>2013-01-07T08:09:22</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/ResponseFileName“>
                <xmsg:Value> EDUA007779</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/PayloadSize“>
                <xmsg:Value>123698B</xmsg:Value>
            </xmsg:Property>
            <xmsg:Property name=”http://www.extra-standard.de/property/State“>
                <xmsg:Value>http://www.extra-
                    standard.de/property/requeststate/FETCHED</xmsg:Value>
            </xmsg:Property>
        </xmsg:Entry>

</xmsg:ListResponse>
```

Hinweis: Da es beim angefragten Abholstatus FETCHED oder CONFIRMED bereits einen **DataRequest** Aufruf gegeben hat, gibt es beim Element **Entry** sowohl den Eintrag einer RequestID als auch einer ResponseID.

3.2.3. Die erweiterte Standardnachricht „DataRequest“

3.2.3.1. Gestaltung der erweiterten Standardnachricht „DataRequest“ V1.3

Die Motivation, die Standardnachricht DataRequest zu erweitern ergibt sich aus den Kritikpunkten, die oben unter 3.2 aufgeführt sind:

- Die Unwägbarkeit des Bearbeitungsaufwands eines DataRequest auf Empfängerseite bzw. der nicht vorhersehbare Umfang der zugehörigen eXTra-Response auf beiden Seiten.
- Die bislang fehlende Möglichkeit, bereits abgeholte und bestätigte fachliche Nachrichten bzw. Daten erneut abholen zu können.

DataRequest wird deshalb erweitert, erhält die nächste Versionsnummer 1.3 sowie den gleichen formalen Aufbau wie die Standardnachricht **ListRequest.**

Neben den drei bekannten Attributen @property, @event und @type wird das Element **Argument** um das Attribut @order erweitert, mit dem die Sortierreihenfolge der angeforderten fachlichen Nachrichten/Rückmeldungen bzw. Daten festgelegt werden kann.

Das Attribut @property gibt unverändert eine abzufragende Eigenschaft an, wobei die Codeliste DataRequestPropertyNames erweitert wird und identisch zur Codeliste ListRequestPropertyNames definiert wird.

Codeliste DataRequestPropertyNames

	xmsg:DataRequestPropertyNamesType
Inhalt	Identifikatoren abfragbare Eigenschaften
Datentyp	`xsd:anyURI`
Individualisierbar	Nein
Beliebige Domain	Nein

Vordefinierte Werte:

```
http://www.extra-standard.de/property/SenderID
http://www.extra-standard.de/property/ReceiverID
http://www.extra-standard.de/property/Procedure
http://www.extra-standard.de/property/DataType
http://www.extra-standard.de/property/RequestID
http://www.extra-standard.de/property/ResponseID
http://www.extra-standard.de/property/RequestCreationTimeStamp
http://www.extra-standard.de/property/ResponseCreationTimeStamp
```

```
http://www.extra-standard.de/property/RequestFileName
http://www.extra-standard.de/property/ResponseFileName
http://www.extra-standard.de/property/Layer
http://www.extra-standard.de/property/State
```

Hinweis: Nutzt man als Kriterium z.B. @name="ResponseID", so ist damit nicht eindeutig festgelegt, ob damit die ResponseID der Transport-Ebene oder der Paket-Ebene der eXTra-Response gemeint ist. Ursache hierfür ist die Auslieferung von fachlichen Nachrichten bzw. Daten auf Grund eines DataRequest in Form von eXTra-Packages oder eXTra-Messages (siehe oben Hinweis zu DataRequest). Mit dem Kriterium @layer kann die erforderliche Eindeutigkeit erzielt werden.

3.2.4. Beispiel für das Zusammenspiel der Standardnachrichten „ListRequest", „ListResponse" und erweitertem „DataRequest"

Mit den beiden Standardnachrichten ListRequest und der Antwort ListResponse erhält der Sender alle Informationen, um ein genaues Bild über den aktuellen Zustand auf Empfängerseite zu gewinnen. Diese Informationen kann der Sender z.B. dazu benutzen, um genau den gewünschten Umfang des mittels erweiterten DataRequest-Aufrufs angestoßenen Holprozess festzulegen. Das folgende Beispiel zeigt die Ablaufsequenz. Der Sender erkundigt sich zunächst über die inzwischen beim Empfänger aufgelaufenen fachlichen Nachrichten/Rückmeldungen bzw. Daten, die der Sender z.B. noch nicht abgeholt hat. Da der Sender in diesem Beispiel bei dem nachfolgenden DataRequest Aufruf eine maximale summarische Datenmenge von 10 MB erhalten will, wertet er die im erhaltenen Inhaltsverzeichnis befindliche Datengröße aus und kann den folgenden DataRequest so zusammen stellen, dass die angeforderten fachlichen Nachrichten/Rückmeldungen bzw. Daten den Umfang von 10 MB nicht überschreiten werden. Als zusätzliche Anforderung will der Sender die beim folgenden DataRequest erhaltenen fachlichen Nachrichten/Rückmeldungen streng in zeitlicher Reihenfolge (die ältesten zuerst) abarbeiten

```
<xmsg:ListRequest version="1.0">
    <xmsg:Query>
        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>12347259</xmsg:EQ>
        </xmsg:Argument>
```

```
        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:EQ>Meldung</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/State" type="xs:string">
                <xmsg:EQ>http://www.extra-standard.de/requeststate/AVAILABLE </xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseFileName"
          event=”RequestData” type="xs:string" >
                <xmsg:GT>EDUA000777</xmsg:GT>
        </xmsg:Argument>

    </xmsg:Query>

</xmsg:ListRequest>
```

Wenn das Zusammenspiel zwischen eXTra-Delivery-Server und Fachverfahren so gestaltet ist, dass zur Auslieferung bereite, aber noch nicht abgeholte fachliche Nachrichten/Rückmeldungen bzw. Daten noch keine ResponseID erhalten haben, dann gibt es im Inhaltsverzeichenis keine Informationen zur RequestID und zur ResponseID.

Angenommen, die Antwort in Form einer **ListResponse** (das Inhaltsverzeichnis) enthält zwölf Einträge mit einem summarischen Datenvolumen von über 20 MB:

```
<xmsg:ListResponse version=“1.0“>

        <xmsg:Entry>
                <xmsg:Property name=”http://www.extra-standard.de/property/SenderID“>
                        <xmsg:Value>11112222</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ReceiverID“>
                        <xmsg:Value>12347259</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/Procedure“>
                        <xmsg:Value>DUA</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/DataType“>
                        <xmsg:Value>Meldung</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-
                        standard.de/property/ResponseCreationTimestamp“>
                        <xmsg:Value>2013-01-17T07:39:50</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ResponseFileName“>
```

```
                <xmsg:Value> EDUA000778</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/PayloadSize“>
                <xmsg:Value>2234567B</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/State“>
                <xmsg:Value>http://www.extra-
                        standard.de/property/requeststate/AVAILABLE</xmsg:Value>
        </xmsg:Property>
</xmsg:Entry>

<xmsg:Entry>
        <xmsg:Property name=”http://www.extra-standard.de/property/SenderID“>
                <xmsg:Value>11112222</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/ReceiverID“>
                <xmsg:Value>12347259</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/Procedure“>
                <xmsg:Value>DUA</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/DataType“>
                <xmsg:Value>Meldung</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-
                standard.de/property/ResponseCreationTimestamp“>
                <xmsg:Value>2013-01-17T13:09:40</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/ResponseFileName“>
                <xmsg:Value> EDUA000779</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/PayloadSize“>
                <xmsg:Value>7123456B</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/State“>
                <xmsg:Value>http://www.extra-
                        standard.de/property/requeststate/AVAILABLE</xmsg:Value>
        </xmsg:Property>
</xmsg:Entry>

<xmsg:Entry>
        <xmsg:Property name=”http://www.extra-standard.de/property/SenderID“>
                <xmsg:Value>11112222</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/ReceiverID“>
                <xmsg:Value>12347259</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/Procedure“>
                <xmsg:Value>DUA</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-standard.de/property/DataType“>
                <xmsg:Value>Meldung</xmsg:Value>
        </xmsg:Property>
        <xmsg:Property name=”http://www.extra-
                standard.de/property/ResponseCreationTimestamp“>
```

```
                    <xmsg:Value>2013-01-17T17:11:22</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ResponseFileName“>
                    <xmsg:Value> EDUA000780</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/PayloadSize“>
                    <xmsg:Value>1234567B</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/State“>
                    <xmsg:Value>http://www.extra-
                        standard.de/property/requeststate/AVAILABLE</xmsg:Value>
                </xmsg:Property>
        </xmsg:Entry>


……<!-- weitere 8 Entry alle mit einem Datenvolumen grösser als 1 MB  -->…….

……<!-- das letzte Entry hat lediglich ein Datenvolumen von ca. 11 KB
        wuerde also zusammen mit den ersten beiden Entry zusammen
        weniger als 10 MB summarisches Datenvolumen ergeben  -->…….


        <xmsg:Entry>
                <xmsg:Property name=”http://www.extra-standard.de/property/SenderID“>
                    <xmsg:Value>11112222</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ReceiverID“>
                    <xmsg:Value>12347259</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/Procedure“>
                    <xmsg:Value>DUA</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/DataType“>
                    <xmsg:Value>Meldung</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-
                    standard.de/property/ResponseCreationTimestamp“>
                    <xmsg:Value>2013-01-18T04:30:10</msg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/ResponseFileName“>
                    <xmsg:Value> EDUA000789</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/PayloadSize“>
                    <xmsg:Value>11822B</xmsg:Value>
                </xmsg:Property>
                <xmsg:Property name=”http://www.extra-standard.de/property/State“>
                    <xmsg:Value>http://www.extra-
                        standard.de/property/requeststate/AVAILABLE</xmsg:Value>
                </xmsg:Property>
        </xmsg:Entry>

</xmsg:ListResponse>
```

Das summarische Datenvolumen der fachlichen Nachrichten/Rückmeldungen bzw. Daten in der zeitlichen Reihenfolge, also der ersten beiden Entry, beträgt ca. 9,3 MB, zusammen mit dem Datenvolumen des folgenden dritten Entry würde die gewünschte Maximalgröße von 10 MB überschritten.

Deshalb ergibt sich für den folgenden gezielten DataRequest-Aufruf folgende Gestalt:

```
<xmsg:DataRequest version=”1.3”>

    <xmsg:Query>

        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>12347259</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:EQ>Meldung</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property=”http://www.extra-standard.de/State”>
                <xmsg:EQ> http://www.extra-standard.de/property/requeststate/AVAILABLE </xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseFileName"
          event=”RequestData” type="xs:string" layer=”Package” order=”ASC”>
                <xmsg:IN>
                        <xmsg:EQ>EDUA000778</xmsg:EQ>
                        <xmsg:EQ>EDUA000779</xmsg:EQ>
                </xmsg:IN>
        < /xmsg:Argument>

    </xmsg:Query>

</xmsg:DataRequest>
```

Hinweis: Dieses Beispiel zeigt auch die unterschiedliche Wirkungsweise die ein Sender zu berücksichtigen hätte, wenn er noch keine ListRequest/ListResponse und noch keinen DataRequest V1.3 zur Verfügung hat. In diesem Fall hätte er lediglich die Möglichkeit gehabt, einen DataRequest V1.2 abzusetzen mit

```
<xmsg:DataRequest version=”1.2”>

    <xmsg:Query>
```

```
        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>12347259</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:EQ>Meldung</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/ResponseFileName"
          event=”RequestData” type="xs:string" state=””>
                <xmsg:GT>EDUA000777</xmsg:GT>
        </xmsg:Argument>

    </xmsg:Query>

    <xmsg:Control>
        <xmsg:MaximumSize>10M</xmsg:MaximumSize>
    </xmsg:Control>


</xmsg:DataRequest>
```

Hinweis: Wenn der Empfänger jetzt nicht prinzipiell die fachlichen Nachrichten/Rückmeldungen bzw. Daten in streng aufsteigender Reihenfolge bzgl. ResponseFileName ausliefert, hätte der Sender in diesem Beispiel womöglich ganz andere Rückmeldungen erhalten, z.B. die Rückmeldungen zu EDUA000778, EDUA000779 und EDUA000789, wenn der Empfänger das vorgegebene maximale Datenvolumen möglichst genau ausschöpfen will.

4. eXTra Standardnachrichten zur Unterstützung des automatischen, bedienerlosen Betriebs

4.1. Szenarien zur Unterstützung des automatischen, bedienerlosen Betriebs

Im laufenden Betrieb kann es vorkommen, dass für den Sender nach einem eXTra-Request ein für ihn unbekannter Zustand beim Empfänger eintritt und der Sender keine eXTra-spezifischen Mittel hat, diesen Zustand in Erfahrung zu bringen.

Folgendes Szenario B.1 sei gegeben:
Der Sender hat eine fachliche Nachricht mit einem eXTra-Request der Art scenario=request-with-acknowledgement abgesendet und als Rückantwort keine eXTra-Response, sondern eine Fehlermeldung der DFÜ-Ebene, z.B. einen Timeout der http(s)-Instanz, erhalten. Für den Sender ist jetzt nicht entscheidbar, ob der Empfänger die vollständige Nachricht nicht empfangen und ablegen konnte, der Sender die Nachricht also noch einmal senden kann. Oder ob der Empfänger die Nachricht zwar empfangen und ablegen konnte, aber die eXTra-Response aus welchem Grund auch immer nicht zurückgeben konnte, z.B. weil der Proxy oder die DFÜ-Ebene auf einen Timeout lief. Wenn jetzt der Sender auf Grund der Fehlermeldung die Nachricht nochmals sendet, hat der Empfänger die Nachricht doppelt erhalten bzw. das Fachverfahren (z.B. das DEÜV-Verfahren) lehnt später die Nachricht wegen Doppellieferung (beim DEÜV-Verfahren wegen falscher laufender Dateinummer) ab.

Wenn der Sender jetzt ein eXTra-spezifisches Sprachmittel hätte, um vom Empfänger exakt die damalige Antwort auf seinen eXTra-Request mit scenario=request-with-acknowledgement anfordern und auswerten könnte, dann hätte er die Möglichkeit, auf die Fehlermeldung sofort automatisch (also ohne menschliche Interaktion, z.B. Anruf bei der Hotline), gezielt und korrekt zu reagieren. Ziel ist es deshalb, ein dafür geeignetes eXTra-spezifisches Sprachmittel zur Verfügung zu stellen.

Folgendes Szenario B.2 sei gegeben:
Der Sender hat eine fachliche Nachricht mit einem eXTra-Request der Art scenario=request-with-response abgesendet und als Rückantwort keine eXTra-Response, sondern eine Fehlermeldung der DFÜ-Ebene, z.B. einen Timeout der http(s)-Instanz, erhalten. Für den

Sender ist jetzt nicht entscheidbar, ob das Fachverfahren beim Empfänger die vollständige Nachricht nicht erhalten und verarbeiten konnte, der Sender die Nachricht also noch einmal senden kann, um die gewünschte Antwort zu erhalten. Oder ob das Fachverfahren beim Empfänger die Nachricht zwar erhalten und verarbeiten konnte, aber die eXTra-Response aus welchem Grund auch immer nicht zurückgeben konnte, z.B. weil der Proxy oder die DFÜ-Ebene auf einen Timeout lief.

Die Frage ist nun, wie die Unsicherheit des Senders zu bewerten ist. Entscheidend hierfür ist, welche Bedeutung der eXTra-Request mit scenario=request-with-response hatte. Wurde mit diesem eXTra-Request eine Auskunftsfunktion des Fachverfahrens auf Empfängerseite angesprochen – also eine Funktion, die den Systemzustand des Fachverfahrens auf Empfängerseite nicht verändert – so kann der Sender den ursprünglichen eXTra-Request einfach wiederholen; dafür ist kein neues eXTra-Sprachmittel erforderlich.

Soll jedoch mit diesem eXTra-Request eine Veränderung des Systemzustands des Fachverfahrens auf Empfängerseite durch eine Verarbeitung der übermittelten fachlichen Daten bewirkt werden, so kann es sein, dass – je nach Ausgestaltung des Fachverfahrens auf Empfängerseite – der Sender auf die nicht erhaltene Response nicht verzichten kann, wenn er weiterhin in jedem Fall korrekte eXTra-Requests senden will bzw. wenn eine Wiederholung des ursprünglichen eXTra-Requests zu einem fehlerhaften Systemzustand des Fachverfahrens auf Empfängerseite führen wurde. In einem derartigen Fall bräuchte der Sender analog zum obigen Szenario B.1 ein eXTra-spezifisches Sprachmittel, um vom Empfänger exakt die damalige Antwort auf seinen eXTra-Request mit scenario=request-with-response anfordern und auswerten zu können.

Auf Grund der Tatsache, dass das Szenario B.2 bei Verwendung der aktuell registrierten verbundspezifischen eXTra-Standards keine Relevanz hat – ein eXTra-Request mit der eXTra-Standardnachricht DataRequest ist als Anforderung einer Auskunft einzustufen, die keine Veränderung des Systemzustands bewirkt bis zum Zeitpunkt, wo eine ConfirmationOfReceipt erfolgt – wird im folgenden lediglich das Szenario B.1 genauer betrachtet.

4.2. Die Standardnachricht „RepeatResponse“

4.2.1. Gestaltung der Standardnachricht „RepeatResponse“

Mit der Standardnachricht RepeatResponse wird die exakte Wiederholung der Antwort auf einen Request mit scenario=request-with-acknowledgement (oder mit scenario=request-with-response) geregelt. Die Wiederholung der Antwort muss hierbei im TransportBody der eXTra-Response zurückgegeben werden, um den Formalismus einer eXTra-Response einzuhalten.

Wenn der ursprüngliche Request des Senders mit scenario=request-with-acknowledgement nur aus der Transportebene bestand, muss die Rückmeldung des Empfängers im TransportBody jetzt nur die Response der damaligen Transportebene, also den Response-TransportHeader, enthalten. Allgemein formuliert: Abhängig davon, wie viele Ebenen der ursprüngliche Request des Senders umfasst hat (z.B. Transport- und Paketebene), muss die Rückmeldung des Empfängers im TransportBody ebenso viele Ebenen umfassen (z.B. bei der GKV Response-TransportHeader und n Response-PackageHeader).

Die Standardnachricht **RepeatResponse** kann für die Spezifikation, auf welche Sendung sich die gewünschte zu wiederholende Antwort bezieht, auf die Sprachmittel eines eXTra-Request – der Transportebene - zurückgreifen.
Analoges gilt auch für die eXTra-Response im TransportBody als Antwort auf einen eXTra-Request mittels **RepeatResponse:** Auch hier stehen bereits alle erforderlichen Sprachmittel aus dem eXTra Basisstandard eXTra Transport zur Verfügung.

Laut Definition ist ein eXTra-Request eineindeutig über die SenderID zusammen mit der RequestID und evtl. dem Timestamp im TransportHeader identifizierbar – egal, aus wie viel Ebenen der eXTra-Request besteht. Deshalb genügt es auch, wenn beim eXTra-Request mittels **RepeatResponse** lediglich die damalige Transportebene, in der Regel nur der damalige TransportHeader, angegeben wird. Sollte ein Sender die SenderID nicht eindeutig belegen, kann er diese Funktion nicht nutzen. Je nach Ausgestaltung des Empfängers kann es evtl. sinnvoll sein, neben dem TransportHeader auch die damaligen PlugIns (z.B. das PlugIn DataSource) anzugeben. Weitere Sprachmittel – eigene Sprachmittel - benötigt die neue Standardnachricht **RepeatResponse** nicht. Es genügt also, den damaligen eXTRa Request-Aufruf mit den Elementen der TransportEbene unter die Elemente Data und ElementSequence

bzw. Base64CharSequence einzuklinken. Dabei ist es ratsam, vom damaligen eXTRa-Request-Aufruf auch das Rootelement Transport (empfohlene Schreibweise ab eXTra 1.3) bzw. XMLTransport (bis eXTra 1.2) einzufügen, um damit auch die Konstellation abfangen zu können, dass zwischen damaligem eXTRa-Request-Aufruf und jetztigen RepeatResponse Aufruf ein eXTra-Versionswechsel stattfand.

Damit ergibt sich für den eXTra-Request mittels **RepeatResponse** im TransportBody folgende Grobstruktur (auf Grund der besseren Lesbarkeit ist der Inhalt von <Base64CharSequence> im Klartext angegeben):

```
<xcpt:Data“>
        <xcpt:Base64CharSequence>
            <xreq:Transport version=”1.3” … >
                <xreq:TransportHeader> ……</xreq:TransportHeader>
                <xreq:TransportPlugIns> ……</xreq:TransportPlugIns>
                 <xreq:TransportBody/>
            </xreq:Transport>
        </xcpt:Base64CharSequence>
</xcpt:Data>
```

Und für die eXTra-Response als Antwort auf einen eXTra-Request mittels **RepeatResponse** ergibt sich im TransportBody folgende Grobstruktur, wenn der ursprüngliche eXTra-request z.B. zwei Ebenen umfasste (die Transport- und die Paketebene; in alternativer Darstellung als <ElementSequence>):

```
<xcpt:Data“>
        <xcpt:ElementSequence>
            <xres:Transport version=”1.3” … >
                <xres:TransportHeader> ……</xres:TransportHeader>
                <xres:TransportPlugIns> ……</xres:TransportPlugIns>
                 <xres:TransportBody>
                    <xres:Package>
                            <xres:PackageHeader> ……</xres:PackageHeader>
                            <xres:PackagePlugIns> ……</xres:PackagePlugIns>
                            <xres:PackageBody/>
                    </xres:Package>
                 </xres:TransportBody>
            </xres:Transport>
        </xcpt:ElementSequence>
</xcpt:Data>
```

Hinweis: Die Besonderheit von RepeatResponse ist dadurch gegeben, dass die Nachricht keine eigenen Sprachmittel benötigt in dem Sinn, dass keine Spezifikation von RepeatResponse in der Schemadatei der eXTra-Standardnachrichten erforderlich ist, weder für den eXTra-Request noch für den eXTra-Response. Aufgrund dessen bleibt die Designentscheidung des eXTra-

Standards erhalten, mit der eine strikte Trennung der Schemadateien für den eXTra-Basisstandard eXTra-Transport von der Schemadatei für die eXTra-Standardnachrichten festgelegt wurde.

4.2.2. Beispiel für die Standardnachricht RepeatResponse

Im folgenden Beispiel wird auch der Fall gezeigt, dass sich die eXTra Version vom ursprünglichen eXTra-Request bis zur späteren Anforderung mittels RepeatResponse von 1.1 auf 1.3 geändert hat.

Annahme: der ursprüngliche eXTra-Request hatte folgende Gestalt:

```
<xreq:XMLTransport version=”1.1” …>
        <xreq:TransportHeader>
            <xcpt:TestIndicator> …</xcpt:TestIndicator>
            <xcpt:Sender> …</xcpt:Sender>
            <xcpt:Receiver> …</xcpt:Receiver>
            <xcpt:RequestDetails>
                                    <xcpt:RequestID class="0">20110228-11112222</xcpt:RequestID>
                                    <xcpt:TimeStamp>2011-02-28T07:39:50</xcpt:TimeStamp>
                                       <xcpt:Application>…</xcpt:Application>

                                       <xcpt:Procedure>http://www.extra-
standard.de/procedures/DEUEV</xcpt:Procedure>
                                       <xcpt:DataType>http://www.extra-
standard.de/datatypes/Sofortmeldung</xcpt:DataType>
                                       <xcpt:Scenario>scenario/request-with-acknowledgement</xcpt:Scenario>
            </xcpt:RequestDetails>
        </xreq:TransportHeader>

        <xreq:TransportPlugIns>
            <xplg:DataSource version="1.0">
                                       <xplg:DataContainer type="http://www.extra-standard.de/container/FILE"
                                       name="EDUA000003" created="2011-02-27T15:08:59" encoding="I8"/>
        …..</xplg:DataSource>
        </xreq:TransportPlugIns>

        <xreq:TransportBody> ….</xreq:TransportBody>
</xreq:XMLTransport>
```

Damit muss die Anforderung zur Wiederholung der Response mittels RepeatResponse folgendermaßen formuliert werden:

```
<xreq:Transport version=”1.3” …>
        <xreq:TransportHeader>
            <xcpt:TestIndicator> …</xcpt:TestIndicator>
            <xcpt:Sender> …</xcpt:Sender>
            <xcpt:Receiver> …</xcpt:Receiver>
            <xcpt:RequestDetails>
                                    <xcpt:RequestID class="0">20110301-11117777</xcpt:RequestID>
                                    <xcpt:TimeStamp>2011-03-01T09:49:51</xcpt:TimeStamp>
                                       <xcpt:Application>…</xcpt:Application>

                                       <xcpt:Procedure>DistributionServer</xcpt:Procedure>
                                       <xcpt:DataType>RepeatResponse</xcpt:DataType>
                                       <xcpt:Scenario>scenario/request-with-response</xcpt:Scenario>
            </xcpt:RequestDetails>
        </xreq:TransportHeader>

        <xreq:TransportPlugIns>
            <xplg:DataTransforms version="1.1">
                    <!— falls das DFÜ-Protokoll http verwendet wird, sollte zur Sicherheit unbedingt die
                         fachlichen Daten im TransportBody verschlüsselt werden
                         => erfordert <Base64CharSequence>; ansonsten genügt <ElementSequence>   -->
        …..</xplg:DataTransforms>
        </xreq:TransportPlugIns>

        <xreq:TransportBody>
            <xcpt:Data>
                <xcpt:ElementSequence>
                    <xreq:XMLTransport version=”1.1” ….>
                        <xreq:TransportHeader>   <!— urspruenglicher TransportHeader -->
                                    <xcpt:TestIndicator> …</xcpt:TestIndicator>
                                    <xcpt:Sender> …</xcpt:Sender>
                                    <xcpt:Receiver> …</xcpt:Receiver>
                                    <xcpt:RequestDetails>
                                       <xcpt:RequestID class="0">20110228-11112222</xcpt:RequestID>
                                       <xcpt:TimeStamp>2011-02-28T07:39:50</xcpt:TimeStamp>
                                       <xcpt:Application>…</xcpt:Application>
                                       <xcpt:Procedure>http://www.extra-
standard.de/procedures/DEUEV</xcpt:Procedure>
                                       <xcpt:DataType>http://www.extra-
standard.de/datatypes/Sofortmeldung</xcpt:DataType>
                                       <xcpt:Scenario>scenario/request-with-acknowledgement</xcpt:Scenario>
                                          </xcpt:RequestDetails>
                        </xreq:TransportHeader>

                        <xreq:TransportPlugIns>
                                     <xplg:DataSource version="1.0">
                                       <xplg:DataContainer type="…" name="EDUA000003" … />
                                    </xplg:DataSource>
                        </xreq:TransportPlugIns>
                        <xreq:TransportBody/>
                    </xreq:XMLTransport>
                </xcpt:ElementSequence>
            </xcpt:Data>
        </xreq:TransportBody>
```

</xreq:Transport>

Die Response – die Antwort – auf die Standardnachricht RepeatResponse sieht z.B. folgendermaßen aus:

```
<xres:Transport version=”1.3” …>
        <xres:TransportHeader>
            <xcpt:TestIndicator> …</xcpt:TestIndicator>
            <xcpt:Sender> …</xcpt:Sender>
            <xcpt:Receiver> …</xcpt:Receiver>
            <xcpt:RequestDetails> … <!-- neuer Request: neue Angaben der RequestDetails -->
                                    <xcpt:RequestID class="0">20110301-11117777</xcpt:RequestID>
                                    <xcpt:TimeStamp>2011-03-01T09:49:51</xcpt:TimeStamp>
                                       <xcpt:Application>…</xcpt:Application>

                                       <xcpt:Procedure>DistributionServer</xcpt:Procedure>
                                       <xcpt:DataType>RepeatResponse</xcpt:DataType>
                                       <xcpt:Scenario>scenario/request-with-response</xcpt:Scenario>
            </xcpt:RequestDetails>
            <xcpt:ResponseDetails> … <!-- direkte Antwort auf den neuen Request -->
                                    <xcpt:ResponseID>…</xcpt:ResponseID>
                                    <xcpt:TimeStamp>2011-03-01-19T09:50:07Z</xcpt:TimeStamp>
                                    <xcpt:Report highestWeight="http://www extra-standard de/weight/OK">
                                            <xcpt:Flag weight="http://www extra-standard de/weight/OK">
                                                    <xcpt:Code>T000</xcpt:Code>
                                                    <xcpt:Text>Alles OK</xcpt:Text>
                                            </xcpt:Flag>
                                    </xcpt:Report>
                                </xcpt:ResponseDetails>
        </xres:TransportHeader>
        <xres:TransportPlugIns>
            <xplg:DataTransforms version="1.1"> ….</xplg:DataTransforms>
        </xres:TransportPlugIns>
        <xres:TransportBody>
            <xcpt:Data>
                <xcpt:ElementSequence>
                        <xres:XMLTransport version=”1.1” ….>
                            <xres:TransportHeader>
                            <!-- hier kommt der TransportHeader wie er beim Sender hätte ankommen sollen→
                                    <xcpt:TestIndicator> …</xcpt:TestIndicator>
                                    <xcpt:Sender> …</xcpt:Sender>
                                    <xcpt:Receiver> …</xcpt:Receiver>
                                    <xcpt:RequestDetails>
                                        <xcpt:RequestID class="0">20110228-11112222</xcpt:RequestID>
                                        <xcpt:TimeStamp>2011-02-28T07:39:50Z</xcpt:TimeStamp>
                                        <xcpt:Application>…</xcpt:Application>
                                       <xcpt:Procedure>http://www.extra-
standard.de/procedures/DEUEV</xcpt:Procedure>
                                    <xcpt:DataType>http://www.extra-
standard.de/datatypes/Sofortmeldung</xcpt:DataType>
                                        <xcpt:Scenario>scenario/request-with-
acknowledgement</xcpt:Scenario>
                                    </xcpt:RequestDetails>
                                    <xcpt:ResponseDetails>
                                    …..<xcpt:ResponseID>…</xcpt:ResponseID>
                                    …..<xcpt:TimeStamp>2011-02-28-T07:40:01Z</xcpt:TimeStamp>
```

```
                                    <xcpt:Report highestWeight="http://www extra-standard de/weight/OK">
                                        <xcpt:Flag weight="http://www extra-standard de/weight/OK">
                                            <xcpt:Code>T000</xcpt:Code>
                                            <xcpt:Text>Alles OK</xcpt:Text>
                                        </xcpt:Flag>
                                    </xcpt:Report>
                                </xcpt:ResponseDetails>
        ……..            </xres:TransportHeader>
        ……..            </xres:TransportBody>
                    </xres:XMLTransport>
                </xcpt:ElementSequence>
            </xcpt:Data>
        <xres:TransportBody/>
</xres:Transport>
```

Die Antwort zeigt in diesem Beispiel, dass die ursprüngliche Datensendung korrekt angenommen und weitergeleitet wurde, siehe Report, Flag, Code und Text „Alles OK“ in den ResponseDetails in der wiederholten Response, aber letztendlich die damalige Response nicht beim Sender ankam.

5. eXTra Standardnachrichten zur Unterstützung des Nachvollzugs

5.1. Szenarien zur Unterstützung von Recherchen und des Nachvollzugs

Die Anforderung eines Senders zu einer Recherche bzw. zum Nachvollzug von Vorgängen

Die Idee ist, eXTra-spezifische Sprachmittel zur Verfügung zu stellen, mit denen Recherchen zu Vorgängen, die in der Vergangenheit liegen, ermöglicht werden. Damit sollen Fragestellungen, wie z.B. „was ist mit meiner Sendung mit der RequestID/ResponseID =xyz vor drei Wochen geworden, weil dafür die Verarbeitungsrück-meldung überfällig ist“, technisch formuliert werden können.

Analoges gilt für eine effektive Unterstützung für den Nachvollzug und die Diagnose insbesondere seltener, ungeklärter oder rätselhafter Vorgänge. Bei der obigen Fragestellung überfälliger Verarbeitungsrückmeldungen wäre es für beide Seite sehr hilfreich, wenn der Sender im Fehlerfall zusätzlich die Stelle, an der die Verarbeitungskette beim Empfänger abbrach, genau fixieren und zudem den Grund des Abbruchs in Erfahrung bringen könnte – ohne auf Empfängerseite langwierig auf die Suche in diversen (Log-) Protokollen gehen zu müssen. In diesem Fall ist die Motivation eine Reduzierung des Aufwands für den Sender ebenso wie für den Empfänger und dessen Hotline.
Auch eine Hilfestellung für Sender wäre damit möglich, welche aufgrund eines technischen Defekts oder sonstigen Katastrophe einen Datenverlust erlitten und nun auf einen Sicherungsbestand vor n Tagen zurückgreifen müssen. Ideal wäre es für den Sender, wenn es eXTra-Sprachmittel gäbe, mit denen er in Erfahrung bringen könnte, was sich sowohl im Sende- wie im Holbetrieb seit seiner letzten Datensicherung alles ergeben hat (siehe auch das Szenario für ListRequest und ListResponse unter 3.2).

Hier geht es also um die Klärung von Vorgängen, die nicht aktuell, sondern z.T. länger in der Vergangenheit zurück liegen.

Eine weitere Überlegung führt ebenfalls dazu, eXTra-Sprachmittel für diesen Fall zur Verfügung zu stellen. Wenn mit der Übermittlung fachlicher Nachrichten nicht nur die technische

Verarbeitung durch das Fachverfahren angestoßen, sondern sich darüber hinaus eine evtl. langwierige Begutachtung, z.B. durch einen menschlichen Prüfer, anschließt, wäre es sehr wünschenswert, wenn es eine Möglichkeit gäbe, sich über den aktuellen Stand der Bearbeitung durch das Fachverfahren oder den menschlichen Bearbeiter zu erkunden.

Diese Anforderungen führen zu Überlegungen, entsprechende eXTra-Standardnachrichten – StatusRequest und als Antwort darauf StatusResponse – zur Verfügung zu stellen.

Rückmeldung des Empfängers auf ein StatusRequest, also auf eine Anforderung zu einer Recherche

Prinzipiell könnte man es dem Empfänger überlassen, wie er auf die Anforderung eines StatusRequest reagiert und welche Informationen er zurückmeldet. Im Sinne einer Standardisierung von Vorgängen und deren Bearbeitung ist jedoch der Ansatz vorzuziehen, auch für die Rückmeldung eine weitere eXTra-Standardnachricht – StatusResponse - zu entwickeln.

Der Empfänger meldet als Antwort auf einen StatusRequest den zuletzt erreichten Zustand der betroffenen Sendungen zurück. Je nachdem, wie die Empfängerseite ausgestaltet ist, kann der zuletzt erreichte Zustand durch

a) den eXTra-Distribution-Server im Minimalausbau => Annahme und Weiterleitung,
b) das Fachverfahren => Verarbeitung oder
c) sogar den eXTra-Delivery-Server im Endausbau => Bereitstellung der Verarbeitungsergebnisse

gemeldet werden.

Im Fall a) im Minimalausbau, wenn nur der eXTra-Distribution-Server den zuletzt erreichten Zustand melden kann, lautet im fehlerfreien Normalfall die Antwort sinngemäß „eXTra-Distribution-Server: Weitergabe an das Fachverfahren erfolgreich durchgeführt“. Im Fehlerfall könnte die Antwort z.B. „eXTra-Distribution-Server: Fehler beim Versuch der Weitergabe an das Fachverfahren“ lauten.

Im Fall b), wenn das Fachverfahren in die Rückmeldung zu einem StatusRequest integriert ist, lautet im fehlerfreien Normalfall die Antwort sinngemäß z.B. „Fachverfahren: Daten erfolgreich

erhalten“ oder „Fachverfahren: Daten erfolgreich verarbeitet“. Im Fehlerfall könnte die Antwort z.B. „Fachverfahren: Fehler beim Verarbeiten der Daten“ lauten.

Im Fall c), wenn der Weg der gesendeten fachlichen Daten über das Fachverfahren bis hin zum eXTra-Delivery-Server verfolgt werden kann, lautet im fehlerfreien Normalfall die Antwort sinngemäß z.B. „eXTra-Delivery-Server: Fachliche Nachricht (Rückmeldung) bzw. Daten vom Fachverfahren erfolgreich erhalten“ (das entspricht dem Zustand @state=“AVAILABLE“ bei der Standardnachricht ListRequest oder ListResponse) oder „eXTra-Delivery-Server: Fachliche Nachricht (Rückmeldung), bzw. Daten erfolgreich ausgeliefert“ (das entspricht dem Zustand @state=“FETCHED“ bei der Standardnachricht ListRequest oder ListResponse) oder „eXTra-Delivery-Server: Bestätigung für fachliche Nachricht (Rückmeldung) bzw. Daten erhalten“ (das entspricht dem Zustand @state=“CONFIRMED“ bei der Standardnachricht ListRequest oder ListResponse). Im Fehlerfall könnte die Antwort z.B. lauten: „eXTra Delivery-Server: Letzte Bestätigung mittels ConfirmationOfReceipt fehlerhaft“.

5.2. Die eXTra Standardnachricht „StatusRequest“

5.2.1. Gestaltung der eXTra Standardnachricht „StatusRequest“

Die Standardnachricht **StatusRequest** kann für die Spezifikation, auf welche Sendung(en) sich die gewünschte Auskunft bezieht, auf die Sprachmittel der Standardnachricht DataRequest und dessen Element **Query** zurückgreifen. Da man mit **StatusRequest** auch eine Menge von Stati anfordern kann (z.B. mit <Argument property=“RequestID“> und <GT>), ist es sinnvoll, die Menge an zurück zu liefernden Stati begrenzen zu können. Analog zur Standardnachricht **DataRequest** kann hierfür das Element **Control** eingesetzt werden.

Damit ergibt sich folgende Grobstruktur

```
<StatusRequest version=“1.1“>
        <Query> ……</Query>
        <Control> …..</Control>
</StatusRequest>
```

Das Element Query

Das Element Query kann analog zur Standardnachricht **DataRequest** eine Folge von n Kindelementen **Argument** enthalten (siehe 3.1.1 und 3.2.3). **Argument** selbst kennt die vier Attribute @event, @property, @type und @order und hat als alternative Kindelemente EQ, GE, LE, GT, LT.

Die Attribute @event, @type und @order sind bzgl. Verwendung und Bedeutung identisch zur Standardnachricht **DataRequest** (siehe 3.2.3) definiert.

Die Codeliste StatusRequestPropertyNames wurde – genauso wie die Codeliste ListRequestPropertyNames (siehe 3.2.1) - gegenüber der Codeliste DataRequestPropertyNames erweitert, um die Begriffe des Senders verwenden zu können und – besonders für die Verfahren der GKV und der Rentenversicherung – den Dateinamen im PlugIn DataSource als Argument verwenden zu können. Damit ergibt sich für die Codeliste StatusRequestPropertyNames:

Codeliste StatusRequestPropertyNames

	xmsg:StatusRequestPropertyNamesType
Inhalt	Identifikatoren abfragbare Eigenschaften
Datentyp	`xsd:anyURI`
Individualisierbar	Nein
Beliebige Domain	Nein

Vordefinierte Werte:

```
http://www.extra-standard.de/property/SenderID
http://www.extra-standard.de/property/ReceiverID
http://www.extra-standard.de/property/Procedure
http://www.extra-standard.de/property/DataType
http://www.extra-standard.de/property/RequestID
http://www.extra-standard.de/property/ResponseID
http://www.extra-standard.de/property/RequestCreationTimeStamp
http://www.extra-standard.de/property/ResponseCreationTimeStamp
http://www.extra-standard.de/property/RequestFileName
http://www.extra-standard.de/property/ResponseFileName
http://www.extra-standard.de/property/Layer
```

Das Element Control

Das Element **Control** wird analog zur Standardnachricht **DataRequest** definiert, jedoch an den Kontext der Standardnachricht **StatusRequest** mit dem Kindelement **MaximumResults** angepasst. Alternativ zu MaximumResults kann auch MaximumSize verwendet werden.

5.2.2. Beispiel für die Standardnachricht „StatusRequest“

Es werden die Stati aller Lieferungen/Packages/Messages angefordert, die der angegebenen Selektion entsprechen.
Argument/@property gibt die abzufragende Eigenschaft an. Das Kindelement EQ, GT usw. spezifiziert Vergleichsoperator und Vergleichswert.

RequestID/ ResponseID bzw. RequestFileName/ ResponseFileName sind Alternativen, um den Range der ursprünglichen Lieferungen präzise einzugrenzen, von denen man den aktuellen Status anfordert.

Voraussetzung für die Nutzung der RequestID/ResponseID/RequestFileName/ ResponseFileName für die Spezifikation einer Menge ursprünglicher Sendungen ist, dass der bzw. die Begriffe streng aufsteigend eineindeutig vergeben werden.

```
<xmsg:StatusRequest version=”1.1”>

    <xmsg:Query>

        <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:EQ>7259</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:EQ>DUA</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:EQ>Meldung</xmsg:EQ>
        </xmsg:Argument>

        <xmsg:Argument property="http://www.extra-standard.de/property/RequestID" type="xs:string"
                        order=”ASC”>
                <xmsg:GT>009999</xmsg:GT>
        </xmsg:Argument>

    </xmsg:Query>
```

```
    <xmsg:Control>
        <xmsg:MaximumResults>100</xmsg:MaximumResults>
        <!--  Begrenzung der Anzahl zurueck zu meldender Stati auf 100 … →
    </xmsg:Control>

</xmsg:StatusRequest>
```

Alternativen zu RequestID sind: ResponseID, RequestFileName oder ResponseFileName.

Hinweis: Ein Begriff aus der Sphäre des Empfängers, z.B. die ResponseID oder der ResponseFileName ist natürlich nur dann eine Alternative, wenn die eXTra-Response auf den ursprünglichen eXTra-Request beim Sender tatsächlich auch erfolgreich empfangen wurde bzw. wenn das Fachverfahren auf Empfängerseite die ursprünglich gesendete fachliche Nachricht auch tatsächlich erhalten und inzwischen verarbeitet hat.

5.3. Die eXTra Standardnachricht „StatusResponse“

5.3.1. Gestaltung der eXTra Standardnachricht „StatusResponse“

Das Gegenstück zur eXTra-Standardnachricht **StatusRequest** ist die Standardnachricht für Rückmeldungen **StatusResponse.** Sie besteht aus zwei Elementen: Dem Element **Property**, mit dem die ursprüngliche Lieferung identifiziert und dem Element **Trace,** mit dem der Status der ursprünglichen Lieferung genau beschrieben wird.

Damit ergibt sich folgende Grobstruktur für die Standardnachricht **StatusResponse**:

```
<StatusResponse version=“1.0“>
        <Property> ……</Property>
        <Trace>……</Trace>
</StatusResponse>
```

Das Element Property

Das Element **Property** ist von der Standardnachricht ConfirmationOfReceipt her bereits bekannt und hat den gleichen Aufbau; es kann mehrfach vorkommen, um eine leichte Lesbarkeit und Eindeutigkeit der Sendung sicherzustellen.

Property hat demgemäß die drei Attribute @name, @type und @event und kennt als Kindelement nur das Element **Value**.

Damit ergibt sich folgende bekannte Grobstruktur für das Element **Property**

```
<Property name=“…“ type=“…” event=”…“>
        <Value>…..</Value>
</Property>
…..
```

Bei den Attributen @name und @event des Elementes **Property** kann man auf die Codelisten StatusRequestPropertyNames und EventNames zurückgreifen (siehe oben unter 3.3.1).

Das Element Trace

Das Element **Trace** selbst hat nur ein Kindelement, nämlich **Checkpoint. Checkpoint** kann mehrfach auftreten, wenn die durchlaufene Prozesskette auf Empfängerseite aufgezeigt werden soll und die Prozesskette aus mehreren Prozessschritten besteht.

Damit ergibt sich für den allgemeinen Fall folgende Grobstruktur für das Element **Trace**

```
<Trace>
	<Checkpoint>
		<Layer>…</Layer>
		<Timestamp>…</Timestamp>
		<Status>…</Status>
		<Report highestWeight="….”t>
			<Flag weight="….">
				<Code>…</Code>
				<Text>...</Text>
			</Flag>
		</Report>
	</Checkpoint>
</Trace>
```

Das Element **Checkpoint** hat vierinsgesamt Kindelemente, nämlich die drei Pflichtelemente **Layer**, **Timestamp** und **Status** sowie das optionale Element **Report.**

Das Element Checkpoint

Die zugelassenen Werte des Kindelementes **Layer** sind wie folgt:

Codeliste LayerNames

	xmsg:LayerNamesType
Inhalt	Identifikatoren abfragbare Eigenschaften
Datentyp	`xsd:anyURI`
Individualisierbar	Ja
Beliebige Domain	Nein

Vordefinierte Werte:

```
http://www.extra-standard.de/layer/Transport
http://www.extra-standard.de/layer/Package
http://www.extra-standard.de/layer/Message
http://www.extra-standard.de/layer/Application
http://www.extra-standard.de/layer/Delivery
```

Die vordefinierten Werte für das Element **Layer** sind bereits für einen möglichen Vollausbau ausgelegt, bei dem auch Stati in Erfahrung gebracht werden können, die das Gesamtverfahren umfasst, bestehend aus Sendeprozess, Fachverfahren, Abhol- und Bestätigungsprozess, bzw. die das betroffene Fachverfahren definiert (über eine Individualisierung des vordefinierten Wertes für Layer/Application).

Eine erste einfache Ausbaustufe könnte z.B. lediglich Stati zurückmelden, die nur den Sendeprozess umfassen (z.B. weil das zugehörige Fachverfahren noch keine Stati zum Ver- bzw. Bearbeitungsstand einer gesendeten Lieferung melden kann). Im Rahmen des Profilierungsvorgangs könnte die Standardnachricht StatusResponse entsprechend profiliert werden, indem die Codeliste für LayerNames nur die Werte für Transport (und ggf. für Package und Message) enthält.

Das Element **Timestamp** ist bezüglich der zugelassenen Werte so definiert wie das Element **Timestamp** des eXTra-Basisschemas, nämlich als Datentyp DateTime.

Für das Element **Status** wird folgende Codeliste definiert:

Codeliste StatusNames

	`xcode:StatusNamesType`
Inhalt	Identifikatoren für einen Zustand auf Empfängerseite
Datentyp	`xsd:anyURI`
Individualisierbar	Ja
Beliebige Domain	Nein

Vordefinierte Werte:

```
http://www.extra-standard.de/status/ACCEPTED
http://www.extra-standard.de/status/PROCESSING
http://www.extra-standard.de/status/COMPLETED
http://www.extra-standard.de/status/CONFIRMED
http://www.extra-standard.de/status/FAILED
```

Die vordefinierten Werte für das Element **Status** sind bereits für einen möglichen Vollausbau ausgelegt, die das Gesamtverfahren umfasst, bestehend aus Sendeprozess, Fachverfahren, Abhol- und Bestätigungsprozess, und bei dem auch differenzierte Bearbeitungsstufen eines

Prozessschrittes (einer eXTra-Ebene bzw. des Fachverfahrens) in Erfahrung gebracht werden können.

Eine erste Ausbaustufe könnte z.B. lediglich Bearbeitungsstufen innerhalb von eXTra-Ebenen zurückmelden, die nur beim Sendeprozess auf Empfängerseite durchlaufen werden (z.B., weil das zugehörige Fachverfahren keine Verbindung herstellen kann zur ursprünglicher Lieferung mit den entsprechenden eXTra-Begriffen RequestID/ResponseID oder RequestFileName). Im Rahmen des Profilierungsvorgangs könnte die Standardnachricht StatusResponse entsprechend profiliert werden, indem die Codeliste für StatusNames nur die Werte für ACCEPTED, PROCESSING, COMPLETED und FAILED enthält.

Die Bedeutung der verschiedenen Ausprägungen ist wie folgt:

- „accepted“: Die übermittelten Daten sind auf der entsprechenden eXTra-Ebene bzw. beim Fachverfahren (siehe **Layer**) angekommen und zumindest in die lokale Datenhaltung übernommen worden. Weitere Arbeitsschritte, wie z.B. Validierung, Komprimieren/Dekomprimieren, Verschlüsseln/Entschlüsseln, Signieren/Signatur prüfen oder verarbeiten sind noch nicht angelaufen. Die Daten sind jedoch definitiv noch nicht an die nächste eXTra-Ebene oder an das Fachverfahren (siehe **Layer**) weitergereicht worden.
- „processing“: Die übermittelten Daten sind auf der entsprechenden eXTra-Ebene bzw. beim Fachverfahren (siehe **Layer**) angekommen und werden dort gerade be- oder verarbeitet. Die notwendigen Arbeitsschritte auf einer eXTra-Ebene, wie z.B. Validierung, Komprimieren/Dekomprimieren, Verschlüsseln/Entschlüsseln, Signieren/Signatur prüfen usw. bzw. das bearbeiten durch das Fachverfahren werden gerade durchlaufen, Fehler sind bisher nicht aufgetreten. Die Daten wurden noch nicht an die nächste eXTra-Ebene bzw. an das Fachverfahren (siehe **Layer**) weitergereicht.
- „completed“: Die jeweilige eXTra-Ebene oder das Fachverfahren hat die Be- oder Verarbeitung erfolgreich abgeschlossen und die Daten erfolgreich an die nächste eXTra-Ebene oder an das Fachverfahren weitergereicht, bzw. das Fachverfahren hat die Daten erfolgreich verarbeitet und die dazugehörige Ergebnismeldung an den eXTra-Delivery Server weitergereicht. Es traten keine Fehler auf.
- „confirmed“: Der Sender hat die dazugehörige Ergebnismeldung mittels DataRequest vom eXTra-Delivery Server angefordert und den Erhalt mittels ConfirmationOfReceipt bestätigt. Es traten keine Fehler auf.

- „failed“ auf der jeweiligen eXTra-Ebene oder im Fachverfahren ist ein Fehler aufgetreten.

Das Element Report

Das Element **Report** und dessen Kindelement **Flag** ist jeweils genauso wie im eXTra Basisstandard eXTra-Transport definiert. **Report** hat demgemäß ein Attribut @**highestWeight, Flag** das Attribut @**weight** und die zwei Kindelemente **Code** und **Text.**

Für das Attribut @**highestWeight** und @**weight** ist somit folgende Codeliste definiert:

Codeliste WeightCode

	xcode:WeightCodeType
Inhalt	Identifikatoren für Fehlergewichte
Datentyp	xsd:anyURI
Individualisierbar	Ja
Beliebige Domain	Nein

Vordefinierte Werte:

```
http://www.extra-standard.de/weight/OK
http://www.extra-standard.de/weight/INFO
http://www.extra-standard.de/weight/WARNING
http://www.extra-standard.de/weight/ERROR
```

5.3.2. Beispiel für die Standardnachricht StatusResponse

Der Sender hat die eXTra-Standardnachricht StatusRequest abgegeben mit der Aufforderung, den Status der abgegebenen Lieferungen mit der RequestID gleich 009999 zurückzugeben. Die Lieferung besteht nur aus der eXTra-Transport-Ebene und die Lieferung wurde vom eXTra-Destribution-Server erfolgreich an das Fachverfahren weitergegeben, aber dort noch nicht weiter verarbeitet bzw. das Fachverfahren wurde in das Gesamtsystem noch nicht integriert (Minimalausbau).

Der TransportBody der Rückmeldung hat hierfür folgende Grobstruktur:

```
<xres:TransportBody>
   <xcpt:Data>
      <xcpt:ElementSeqence>
            <xmsg:StatusResponse> …. </xmsg:StatusResponse>
      </xcpt:ElementSeqence>
```

```
    </xcpt:Data>
</xres:TransportBody>
```

Die StatusResponse Nachricht sieht in diesem Fall z.B. folgendermaßen aus:

```
<xmsg:StatusResponse>

        <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:Value>7259</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:Value>DEUEV</xmsg:Value>
        </xmsg:Propertyt>

        <xmsg:Property name="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:Value>Sofortmeldung</xmsg:Value>
        </xmsg:Argument>

        <xmsg:Property name="http://www.extra-standard.de/property/RequestID" type="xs:string">
                <xmsg:Value>009999</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/DatasourceFilename"
type="xs:string">
                <xmsg:Value>EDUA0000003</xmsg:Value>
        </xmsg:Property>

        <xmsg:Trace
                <msg:Checkpoint>
                        <msg:Layer>http://www.extra-standard.de/level/Transport</msg:Layer>
                        <msg:TimeStamp>2010-10-08T20:10:44</msg:TimeStamp>
                        <msg:Status>http://www.extra-standard.de/state/COMPLETED</msg:Status>
                        <msg:Report highestWeight=http://www.extra-standard.de/weight/INFO>
                                <msg:Flag weight=http://www.extra-standard.de/weight/INFO>
                                        <msg:Code>T1000</msg:Code>
                                        <msg:Text>alles ok</msg:Text>
                                </msg:Flag>
                        </msg:Report>
                </msg:Checkpoint>

        </xmsg:Trace>
</xmsg:StatusResponse>
```

6. Die ListOf-Nachrichten

Für den Sender eröffnen die sogenannten ListOf-Nachrichten die Möglichkeit, mehrere eXTra-Standardnachrichten des gleichen Typs mit einem eXTra-Request abzusenden, um damit die Anzahl der Kommunikationsvorgänge zwischen Sender und Empfänger zu reduzieren. Analoges gilt für den Empfänger, der mit einem eXTra-Response mehrere eXTra-Standardnachrichten des gleichen Typs übermitteln kann.

6.1. Gestaltung der ListOf Nachrichten

Eine ListOf Nachricht stellt lediglich eine Klammer dar für eine Folge zugehöriger Standardnachrichten des Typs **DataRequest**, **ConfirmationOfReceipt** oder **StatusResponse**. Dementsprechend gibt es folgende ListOf Nachrichten: **ListOfDataRequest**, **ListOfConfirmationOfReceipt** und **ListOfStatusResponse**.

Für alle ListOf Nachrichten ergibt sich folgende Grobstruktur, hier beispielhaft aufgezeigt an Hand der Standardnachricht **ListOfConfirmationOfReceipt**

```
<ListOfConfirmationOfReceipt version="1.0">
    <ConfirmationOfReceipt> ……</ConfirmationOfReceipt>
    …..
    <ConfirmationOfReceipt> ……</ConfirmationOfReceipt>
    …..
    <ConfirmationOfReceipt> ……</ConfirmationOfReceipt>
</ListOfConfirmationOfReceipt>
```

Während der Sender die Wahl hat, ob er einen eXTra-Request mit einer ListOf-Nachricht oder mehrere eXTra-Requests mit jeweils einer zugehörigen Standardnachricht, z.B. **DataRequest**, sendet, liegt bei **ListOfStatusResponse** ein anderer Sachverhalt zu Grunde: Wenn der Sender einen eXTra-Request mit der Standardnachricht StatusRequest abgesetzt hat, der den Status mehrerer Lieferungen nachfragt, dann erzeugt der Empfänger einen eXTra-Response, der die Nachricht **ListOfStatusResponse** mit einer Folge von **StatusResponse** mit dem Status jeweils einer der nachgefragten Lieferungen enthält.

6.2. Beispiel für die Standardnachricht ListOfDataRequest

Beispiel im Kontext der gesetzlichen Krankenkassen

Der Sender will in diesem Beispiel receiver- und fachverfahrenspezifisch vorgehen und gleichzeitig die Anzahl der notwendigen Kommunikationsvorgänge reduzieren. Pro eXTra-Request fordert er genau von einer „Datenannahme- und Verteilsttelle“ (DAV) die Rückmeldungen nicht zu allen, sondern gezielt zu zwei Fachverfahren an: Zum DEÜV-Verfahren (DUA) und zum Vergabeverfahren von Versicherungsnummern (VSA). Der eXTra-Request enthält beispielsweise folgenden **ListOfDataRequest** .

```
<xmsg:ListOfDataRequest version=”1.0”>

  <xmsg:DataRequest version=”1.2”>            DataRequest1 an DAV1 mit  DUA

    <xmsg:Query>

      <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
            <xmsg:EQ>1231111</xmsg:EQ>
      </xmsg:Argument>

      <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
            <xmsg:EQ>DUA</xmsg:EQ>
      </xmsg:Argument>

    </xmsg:Query>

  </xmsg:DataRequest>

  <xmsg:DataRequest version=”1.2”>            DataRequest2 an DAV1 mit VSA

    <xmsg:Query>

      <xmsg:Argument property="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
            <xmsg:EQ>1231111</xmsg:EQ>
      </xmsg:Argument>

      <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
            <xmsg:EQ>VSA</xmsg:EQ>
      </xmsg:Argument>

    </xmsg:Query>

  </xmsg:DataRequest>


</xmsg:ListOfDataRequest>
```

6.3. Beispiel für die Standardnachricht ListOfConfirmationOfReceipt

Beispiel im Kontext der gesetzlichen Krankenkassen

Im Beispiel wird gezeigt, wie der Sender die mit der eXTra-Response enthaltenen fachlichen Nachrichten in effizienter Weise bestätigen kann. Das Beispiel zeigt auch exemplarisch, wie unvorhersehbar der Umfang einer eXTra-Response auf einen DataRequest für den Sender sein kann, hier sind es für das Fachverfahren DUA drei Rückmeldungen vom Receiver 1 und zwei vom Receiver 2. Genauso schwer einschätzbar ist für den Empfänger eines DataRequest naturgemäß, wie aufwändig dessen Bearbeitung sein wird.

Folgenden DataRequest hat der Sender zu Beginn der Prozesskette gesendet:

```
<xmsg:DataRequest version=”1.2”>          DataRequest unspezifisch an alle DAVen mit  DUA
  <xmsg:Query>
    <xmsg:Argument property="http://www.extra-standard.de/property/Procedure" type="xs:string">
            <xmsg:EQ>DUA</xmsg:EQ>
    </xmsg:Argument>
  </xmsg:Query>
</xmsg:DataRequest>
```

Hier wird angenommen, dass der Sender in der zum **DataRequest** zugehörigen eXTra-Response nur von zwei Receivern (DAVen) drei bzw. zwei Rückmeldungen erhält.

Hinweis: Insbesondere wenn der Sender relativ selten **DataRequest**-Aufrufe absetzt, könnte die eXTra-Response umfangreicher als gewünscht ausfallen. Um sowohl den Bearbeitungsaufwand auf Empfängerseite wie auch den Umfang der Rückmeldungen für den Sender einigermaßen überschaubar zu halten, ist es deshalb empfehlenswert, dass der Sender nach einem Sendeprozess mit fachlichen Nachrichten nicht zu lange mit einem folgenden **DataRequest** Aufruf wartet.

Hinweis: Können aus Sicht des Senders Verarbeitungsspitzen nicht ausgeschlossen werden, oder schwankt der Zeitraum von einem DataRequest Aufruf zum nächsten stark, dann

empfiehlt es sich, auf die Standardnachrichten **ListRequest**, **ListResponse** und dem erweiterten **DataRequest Version 1.3** zurückzugreifen.

Der Sender bestätigt nun den beiden Receivern (DAVn) die erhaltenen drei bzw. zwei Rückmeldungen mit einem einzigen eXTra-Request mittels folgendem **ListOfConfirmationOfReceipt**.

Hinweis für den Datenübermittlungsverbund der gesetzlichen Krankenkassen GKV: Da im Dateinamen die laufenden Dateinummern nur eindeutig innerhalb einer DAV sind, sollten die jeweiligen ReceiverIDs angegeben werden, um eine eindeutige Zuordnung von Datei zu Receiver (DAV) zu gewährleisten. Für eine derartige „Sammelbestätigung“ sind also implizit mehrere **ConfirmationOfReceipt** innerhalb einer **ListOfConfirmationOfReceipt**-Nachricht erforderlich.

```
<xmsg:ListOfConfirmationOfReceipt version="1.0">

    <xmsg:ConfirmationOfReceipt version="1.3">

        <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID"
                type="xs:string">
                <xmsg:Value>1231111</xmsg:Value>
        </xmsg:Property>


        <xmsg:PropertySet name="http://www.extra-standard.de/property/ResponseFilename"
                type="xs:string">
        <!-- Bestaetigung von 3 Rueckmeldungen jeweils mit dem entsprechenden Dateinamen
        -->
                <xmsg:Value>EDUA0007261</xmsg:Value>
                <xmsg:Value>EDUA0007262</xmsg:Value>
                <xmsg:value>EDUA0007266</xmsg:value>

         </xmsg:PropertySet>

    </xmsg:ConfirmationOfReceipt>



    <xmsg:ConfirmationOfReceipt version="1.3">

        <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID"
                type="xs:string">
                <xmsg:Value>1232222</xmsg:Value>
        </xmsg:Property>
```

```
        <xmsg:PropertySet name="http://www.extra-standard.de/property/ResponseFilename"
                type="xs:string">
        <!-- Bestaetigung von 2 Rueckmeldungen jeweils mit dem entsprechenden Dateinamen
        -->
                <xmsg:Value>EDUA0000567</xmsg:Value>
                <xmsg:Value>EDUA0000568</xmsg:Value>

         </xmsg:PropertySet>

    </xmsg:ConfirmationOfReceipt>

</xmsg:ListOfConfirmationOfReceipt>
```

6.4. Beispiel für die Standardnachricht ListOfStatusResponse

Der Sender hat die eXTra-Standardnachricht StatusRequest abgegeben mit der Aufforderung, den Status aller abgegebenen Lieferungen mit der RequestID größer 009999 zurückzugeben. Folgende Annahmen gelten für dieses Beispiel:

- Die Lieferungen bestehen immer nur aus der eXTra-Transport-Ebene.
- Die Statusanforderung trifft für drei Lieferungen zu.
- Das Gesamtsystem auf Empfängerseite ist vollständig ausgebaut, d.h. eine vom Sender abgegebene Lieferung kann bis zuletzt, bis zur Bestätigung durch den Sender mittels ConfirmationOfReceipt, nachvollzogen werden.
- Die älteste Lieferung (RequestID=009999) wurde vom Fachverfahren verarbeitet und die Rückmeldung an den eXTra-Delivery-Server weitergeben, aber noch nicht mittels DataRequest abgeholt.
- Die nächste Lieferung (RequestID=010000) wurde vom Fachverfahren verarbeitet, aber die Rückmeldung noch nicht an den eXTra-Delivery-Server weitergegeben.
- Die jüngste Lieferung (RequestID=010001) lief noch vor der Weiterleitung an das Fachverfahren auf einen Fehler auf der Transportebene.

Die eXTra-Response besteht aus der ListOfStatusResponse Nachricht, die wiederum aus einer Folge von StatusResponse Nachrichten besteht, in diesem Beispiel aus drei Nachrichten. Der TransportBody hat hierfür folgende Grobstruktur:

```
<xres:TransportBody>
   <xcpt:Data>
      <xcpt:ElementSeqence>
         <xmsg:ListOfStatusResponse version=”1.0”>
            <xmsg:StatusResponse> …. </xmsg:StatusResponse>   <!-- Status zu Lieferung 1 -->
```

```
                <xmsg:StatusResponse> …. </xmsg:StatusResponse>   <!-- Status zu Lieferung 2 -->

                <xmsg:StatusResponse> …. </xmsg:StatusResponse>   <!-- Status zu Lieferung 3 -->
            </xmsg:ListOfStatusResponse>
        </xcpt:ElementSeqence>
    </xcpt:Data>
</xres:TransportBody>
```

Im Detail haben die drei Statusnachrichten z.B. folgende Ausprägung:

Status zu Lieferung 1 mit der RequestID 009999, fehlerfrei

```
<xmsg:StatusResponse>
        <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:Value>7259</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:Value>DEUEV</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:Value>Sofortmeldung</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/RequestID" type="xs:string">
                <xmsg:Value>009999</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/DatasourceFilename"
type="xs:string">
                <xmsg:Value>EDUA0000018</xmsg:Value>
        </xmsg:Property>


        <xmsg:Trace
            <msg:Checkpoint>
                <msg:Layer>http://www.extra-standard.de/level/Delivery</msg:Layer>
                <msg:TimeStamp>2010-10-08T20:10:44</msg:TimeStamp>
                <msg:Status>http://www.extra-standard.de/state/ACCEPTED</msg:Status>
                <msg:Report highestWeight=http://www.extra-standard.de/weight/INFO>
                        <msg:Flag weight=http://www.extra-standard.de/weight/INFO>
                                <msg:Code>D1000</msg:Code>
                                <msg:Text>alles ok</msg:Text>
                        </msg:Flag>
                </msg:Report>
            </msg:Checkpoint>
        </xmsg:Trace>

</xmsg:StatusResponse>
```

Status zu Lieferung 2 mit der RequestID 010000, fehlerfrei

```
<xmsg:StatusResponse>

        <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:Value>7259</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:Value>DEUEV</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:Value>Sofortmeldung</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/RequestID" type="xs:string">
                <xmsg:Value>010000</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/DatasourceFilename"
type="xs:string">
                <xmsg:Value>EDUA0000019</xmsg:Value>
        </xmsg:Property>


        <xmsg:Trace
            <msg:Checkpoint>
                <msg:Layer>http://www.extra-standard.de/level/Application</msg:Layer>
                <msg:TimeStamp>2010-10-09T20:15:14</msg:TimeStamp>
                <msg:Status>http://www.extra-standard.de/state/PROCESSING</msg:Status>
                <msg:Report highestWeight=http://www.extra-standard.de/weight/INFO>
                        <msg:Flag weight=http://www.extra-standard.de/weight/INFO>
                                <msg:Code>A0000</msg:Code>
                                <msg:Text>alles ok</msg:Text>
                        </msg:Flag>
                </msg:Report>
            </msg:Checkpoint>
        </xmsg:Trace>


</xmsg:StatusResponse>
```

Status zu Lieferung 3 mit der RequestID 010001; Fehler auf der Transportebene

```
<xmsg:StatusResponse>

        <xmsg:Property name="http://www.extra-standard.de/property/ReceiverID" type="xs:string">
                <xmsg:Value>7259</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/Procedure" type="xs:string">
                <xmsg:Value>DEUEV</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/Datatype" type="xs:string">
                <xmsg:Value>Sofortmeldung</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/RequestID" type="xs:string">
                <xmsg:Value>010001</xmsg:Value>
        </xmsg:Property>

        <xmsg:Property name="http://www.extra-standard.de/property/DatasourceFilename"
type="xs:string">
                <xmsg:Value>EDUA0000020</xmsg:Value>
        </xmsg:Property>


        <xmsg:Trace
            <msg:Checkpoint>
                <msg:Layer>http://www.extra-standard.de/level/Transport</msg:Layer>
                <msg:TimeStamp>2010-10-09T21:05:04</msg:TimeStamp>
                <msg:Status>http://www.extra-standard.de/state/FAILED</msg:Status>

                <msg:Report highestWeight=http://www.extra-standard.de/weight/ERROR>
                        <msg:Flag weight=http://www.extra-standard.de/weight/ERROR>
                                <msg:Code>TF100</msg:Code>
                                <msg:Text>TransportEbene Fehler: Die Nachricht konnte nicht
                                        entschluesselt werden</msg:Text>
                        </msg:Flag>
                </msg:Report>
            </msg:Checkpoint>
        </xmsg:Trace>


</xmsg:StatusResponse>
```

7. Anhang

7.1. Referenzen

Kurzname	Quelle
BEST	*eXTra Best Practices* zu finden unter www.extra-standard.de
DSIG	*eXTra Design Guidelines*, zu finden unter www.extra-standard.de
EINF	*Einführung in den eXTra-Standard*, zu finden unter www.extra-standard.de
EMSG	*eXTra-Standardnachrichten, Schnittstellenbeschreibung*, zu finden unter www.extra-standard.de
EXSEC	*Sicherheit und Verfügbarkeit in einem eXTra spezifischen Datenübermittlungsverbund*, zu finden unter www.extra-standard.de
EXWS	*eXTra und Webservices*, zu finden unter www.extra-standard.de
IFACE	*eXTra-Transport Schnittstellenbeschreibung*, zu finden unter www.extra-standard.de
KOMP	*eXTra-Kompendium*, zu finden unter www.extra-standard.de
RFC2119	*Request for Comments: 2119*, S. Bradner, Harvard University, March 1997, http://www.ietf.org/rfc/rfc2119.txt
PROF	*eXTra-Profilierung,* zu finden unter www.extra-standard.de
ÜMSG	*eXTra Standardnachrichten, Überblick*, zu finden unter www.extra-standard.de
VERS	*eXTra Versionierung*, zu finden unter www.extra-standard.de
XENC	*XML Encryption*, http://www.w3.org/TR/xmlenc-core/
XML	*XML Recommendation 1.0, 3rd Edition*, http://www.w3.org/XML
XSD	*XML Schema Definition*, http://www.w3.org/TR/xmlschema-0/
XSIG	*XML Signature*, http://www.w3.org/TR/xmldsig-core/
XSL	*XML Stylesheet Language*, http://www.w3.org/TR/1999/REC-xslt-19991116, http://www.w3.org/TR/xslt20/

7.2. Glossar

Begriff	Erklärung
Authentifizierung	Authentifizierung ist der Nachweis (Verifizierung) einer behaupteten Eigenschaft eines Systems, eines Dokumentes oder einer Information. Bei einer Authentifizierung zwischen zwei Parteien authentisiert sich die Eine (z.B. der Sender), während die Andere (z.B. der Empfänger) die Erstere authentifiziert. Eine in der EDV weit verbreitete Form ist die Authentifizierung mittels UserID, Passwort, oder auch mittels Zertifikat und Signatur
Body	Im eXTra-Datenmodell enthält dieser Bereich entweder die gesamte nächsttiefere Ebene oder im Fall der untersten Ebene die fachlichen Nutzdaten.
Clearing-Stelle	Zentrale annehmende Stelle von Daten, welche die Funktion eines Bindegliedes zwischen fachlichem Sender und fachlichem Empfänger darstellt und für beide Seiten in der Regel für Rechtssicherheit und einen geordneten Betrieb sorgt.
Datenübermittlungsverbund	Heute können in einem Datenübermittlungsverbund definierte Nachrichten über ein konkretes Datenübermittlungsverfahren ausgetauscht werden. In der Regel definiert der Empfänger der Daten (zumeist eine Behörde, ein Verband oder eine Institution) sowohl das Datenübermittlungsverfahren als auch die Nachrichten. Die Teilnehmer, die Datenlieferanten, müssen üblicherweise beim Empfänger registriert sein. Die Idee von eXTra ist es für beliebige Datenübermittlungsverbünde ein einheitliches Datenübermittlungsverfahren, bzw. über ein generisches Konzept – der *Profilierung* - eine Familie verwandter Datenübermittlungsverfahren zur Verfügung zu stellen. Das Ergebnis der Profilierung ist ein *verbundspezifischer eXTra-Standard*.
DFÜ Ebene	Die DFÜ Ebene repräsentiert innerhalb des abstrakten Architekturmodells eines Datenübermittlungssystems, sowie bei eXTra eine Ebene, bei der ein *DFÜ Sender* Daten mit einem *DFÜ Protokoll* an einen *DFÜ Empfänger* sendet. Weiterhin sind in der DFÜ Ebene in der Regel Sicherheitsmaßnahmen integriert, die insbesondere Angriffen aus der Internet-Welt entgegen wirken sollen. In eXTra wird die Ausgestaltung der DFÜ Ebene nicht behandelt; insofern trifft eXTra keinerlei Aussagen zum DFÜ-Protokoll oder zu den dort angesiedelten Sicherheitsmaßnahmen.
ELSTER	Die elektronische Steuererklärung ELSTER ist ein Datenübermittlungsverfahren der deutschen Steuerverwaltungen aller Länder und des Bundes mit dem ein Steuerpflichtiger oder in dessen Auftrag ein Dienstleister Erklärungs- und Anmeldesteuern an die Finanzverwaltung übermitteln kann. Auf Wunsch stellt die Finanzverwaltung Bescheide zum Abholen durch den Steuerpflichtigen bzw. in dessen Auftrag einem Dienstleister zum Abholen bereit.
eSTATISTIK.core	Datenübermittlungsverfahren des Statistischen Bundesamtes zur automatischen Übermittlung von statistischen Daten aus betrieblichen Anwendungen der Melder.

Begriff	Erklärung
Fachverfahren	Mit Hilfe von eXTra kann ein Fachverfahren auf Senderseite – der fachliche Sender – fachliche Daten an das zugeordnete Fachverfahren auf Empfängerseite – den fachlichen Empfänger – übermitteln. In umgekehrter Datenflussrichtung kann ein erzeugendes Fachverfahren auf Empfängerseite dem anfordernden Fachverfahren auf Senderseite fachliche Daten zum Abholen bereitstellen.
Header	In der Informationstechnik werden Metadaten am Anfang einer Datei oder eines Datenblocks als Header (auch: Dateikopf) bezeichnet. Diese können verwendet werden, um beispielsweise das Dateiformat zu beschreiben oder weitere Angaben zu den Daten zu machen.
Kommunikationsszenario	Ein Kommunikationsszenario definiert das erwartete Verhalten eines Empfängers auf einer Ebene.
Kommunikationsvorgang	Ein Kommunikationsvorgang definiert den Ablauf der Kommunikation zwischen Sender und Empfänger auf einer Ebene des eXTra-Kommunikationsmodells. Er legt fest, wie die Rollen Sender und Empfänger verteilt werden und ob eine synchrone Response möglich ist.
Komprimierung	Datenkompression oder Datenkomprimierung ist die Anwendung von Verfahren zur Reduktion des Speicherbedarfs von Daten.
Logischer Empfänger	Der logische Empfänger ist ein Akteur, der auf Empfänger-Seite der *Logistikebene* zugeordnet ist.
Logischer Sender	Der logische Sender ist ein Akteur, der auf Sender-Seite der *Logistikebene* zugeordnet ist.
Logistikebene	Die Logistik- oder auch Paketebene repräsentiert bei eXTra eine Ebene, die auf Senderseite durch den *logischen Sender* für die Bündelung mehrerer fachlichen Nachrichten für einen Endempfänger zu einem Paket zuständig ist, bzw. auf Empfängerseite durch den *logischen Empfänger* die Verteilung der Paketinhalte auf die Endempfänger/ Verwerter übernimmt.
Logging	Eine Logdatei beinhaltet das automatisch erstellte Protokoll aller oder bestimmter Aktionen von Prozessen auf einem Computersystem. Das Fortschreiben dieser Datei nennt man Logging.
Message	(Fach-)Nachricht, die ein Erzeuger mithilfe eines Fachverfahrens generiert und die ein Verwerter verarbeiten soll. Eine eXTra-Message enthält im MessageBody in der Regel eine einzige Fachnachricht. Dadurch ist eine Fachnachricht für eXTra sicht- und greifbar.
Migration	Unter Migration versteht man im Rahmen der Informationstechnik den Umstieg eines wesentlichen Teils der eingesetzten Software beziehungsweise den Transfer von Daten aus einer Umgebung in eine andere, sowie die Umstellung von Hardware einer alten Technologie in neue Technologien unter weitgehender Nutzung vorhandener Infrastrukturen.
Nachrichtenebene	Die Nachrichtenebene repräsentiert bei eXTra eine Ebene, bei der z.B. auf Senderseite ein Erzeuger durch ein Fachverfahren eine fachliche

Begriff	Erklärung
	Nachricht generiert, die in Form einer eXTra-Message übermittelt wird. Auf der Empfängerseite verarbeitet das korrespondierende Fachverfahren als Endempfänger und Verwerter die fachliche Nachricht.
Package	Ein Paket, das z.B. ein *logischer Sender* an einen *logischen Empfänger* übermittelt. Ein Package enthält im PackageBody entweder die fachlichen Daten oder die nächsttiefere Ebene, die Nachrichtenebene. Sind im PackageBody fachliche Daten enthalten, so ist es für eXTra nicht erkennbar, aus wievielen fachlichen Nachrichten der PackageBody zusammengesetzt ist.
Paketebene	Die Paket- oder Logistikebene erlaubt es, Einzelnachrichten zu Paketen zusammenzufassen und damit in verschiedener Hinsicht einheitlich zu behandeln. Sie unterstützt damit insbesondere die Massendatenverarbeitung. Ein derartiges Paket wird in Form eines eXTra-*Package* übermittelt.
physikalischer Empfänger	Der physikalische Empfänger ist ein Akteur, der auf Empfänger-Seite der *Transportebene* zugeordnet ist. Physikalischer Sender und –Empfänger stehen über ein konkretes Kommunikationssystem direkt miteinander in Verbindung und tauschen darüber eXTra-Dokumente aus. Die Ausgestaltung des konkreten Kommunikationssystems (und damit der verwendeten DFÜ-Protokolle und Netze) ist in eXTra nicht vorgegeben.
physikalischer Sender	Der physikalische Sender ist ein Akteur, der auf Sender-Seite der *Transportebene* zugeordnet ist. Physikalischer Sender und Empfänger stehen über ein konkretes Kommunikationssystem direkt miteinander in Verbindung und tauschen darüber eXTra-Dokumente aus. Die Ausgestaltung des konkreten Kommunikationssystems (und damit der verwendeten DFÜ-Protokolle und Netze) ist in eXTra nicht vorgegeben.
Plug-Ins	Softwarehersteller definieren Schnittstellen zu ihren Produkten, mit deren Hilfe Dritte Funktionserweiterungen (Plug-Ins) für diese Softwareprodukte programmieren können. In eXTra sind Plug-Ins optionale Erweiterungen des Datenmodells, die aber nicht unabhängig entwickelt werden können, sondern dem Standardisierungsprozess unterliegen
Profilkonfiguration Profilierung	Die Profilkonfiguration ist bei eXTra eine XML-Datei, die dazu dient aus dem allgemeinen eXTra-Basisschema für ein konkretes Fachverfahren bzw. einen konkreten *Datenübermittlungsverbund* auf formale Weise eine spezifische Schemadatei – ein eXTra-Subschema - zu generieren. Diesen Generierungsvorgang – die Profilierung - kann jedes Fachverfahren bzw. jeder Datenübermittlungsverbund selbst durchführen.
Request-ID	Anfragekennung. In der eXTra-Terminologie ist die Request-ID ein vom Sender vergebener eindeutiger Identifikator einer Anfrage.
Response-ID	Antwortkennung In der eXTra-Terminologie ist die Response-ID ein vom Empfänger vergebener eindeutiger Identifikator einer Antwort auf eine Anforderung mit einer eindeutigen RequestID.

Begriff	Erklärung
Signierung	Unter einer elektronischen Signatur versteht man Daten, mit denen man den Unterzeichner bzw. Signaturersteller identifizieren kann und sich die Integrität der signierten, elektronischen Daten prüfen lässt. Die elektronische Signatur erfüllt somit technisch gesehen unter bestimmten Bedingungen den gleichen Zweck wie eine eigenhändige Unterschrift auf Papierdokumenten. Den Vorgang nennt man Signierung.
Topologie	Die Topologie bezeichnet bei einem Computernetz die Struktur der Verbindungen mehrerer Geräte untereinander, um einen gemeinsamen Datenaustausch zu gewährleisten. Bei einem Datenübermittlungsverbund stellt die Topologie die Struktur der Verbindungen der einzelnen Teilnehmer bzw. Systeme des Datenübermittlungsverbundes dar.
Transportebene	Die Transportebene repräsentiert bei eXTra eine Ebene, bei der ein *physikalischer Sender* vollständige eXTra-Dokumente an einen *physikalischen Empfänger* sendet. Im TransportBody sind entweder die fachlichen Daten oder die nächsttiefere Ebene, die Paket- oder Nachrichtenebene enthalten. Sind im TransportBody fachliche Daten enthalten, so ist es für eXTra nicht erkennbar, aus wievielen fachlichen Nachrichten der TransportBody zusammengesetzt ist.
URL	Als Uniform Resource Locator (URL, engl. „einheitlicher Quellenanzeiger“) bezeichnet man eine Unterart von Uniform Resource Identifiern (URIs). URLs identifizieren eine Ressource über das verwendete Netzwerkprotokoll (beispielsweise http oder ftp) und den Ort (engl. location) der Ressource in Computernetzwerken.
UUID	Ein **Universally Unique Identifier** (**UUID**) ist ein Standard für Identifikatoren, der in der Softwareentwicklung verwendet wird. Er ist von der Open Software Foundation (OSF) als Teil des Distributed Computing Environment (DCE) standardisiert. Sinn und Zweck von UUIDs ist, Informationen in verteilten Systemen ohne zentrale Koordination eindeutig kennzeichnen zu können.
Validierung	In der Softwaretechnik bezeichnet Validierung (auch Plausibilisierung, als Test auf Plausibilität, oder engl. Sanity Check genannt) die Kontrolle eines konkreten Wertes darauf, ob er zu einem bestimmten Datentyp gehört oder in einem vorgegebenen Wertebereich oder einer vorgegebenen Wertemenge liegt.
verbundspezifischer eXTra-Standard	Die Idee von eXTra ist es für beliebige Datenübermittlungsverbünde ein einheitliches Datenübermittlungsverfahren, bzw. über ein generisches Konzept – der *Profilierung* - eine Familie verwandter Datenübermittlungsverfahren zur Verfügung zu stellen. Das Ergebnis der Profilierung ist ein verbundspezifischer eXTra-Standard.
Verschlüsselung	Verschlüsselung nennt man den Vorgang, bei dem die Repräsentation einer Informationseinheit wie etwa ein Text oder eine Bilddatei aus einer unverschlüsselten Form, dem sogenannten Klartext, in eine verschlüsselte Form, dem sogenannten Geheimtext, überführt wird. In der Regel erfolgen Ver- und Entschlüsselung mit Hilfe mathematischer Verfahren, die hierzu

Begriff	Erklärung
	ein oder mehrere extern zugeführte Schlüssel verwenden.
W3C	Das World Wide Web Consortium (W3C, http:www.w3.org) entwickelt Standards und Technologien für das Internet
XSLT Stylesheet	XSL Transformation, kurz XSLT, ist eine XML-basierte Sprache für die Transformation von XML-Dokumenten. Sie ist Teil des W3C-Standards Extensible Stylesheet Language (XSL).