Koordinierungsstelle
für IT-Standards

KOORDINIERUNGSSTELLE FÜR IT-STANDARDS (KOSIT)

Bremen

Fortschreibung des Standards OSCI Transport V2.0:

ÜBERSICHT: OSCI TRANSPORT VERSION 2.01

Hintergrund, Erläuterung der Änderungen und Erweiterungen

Februar 2013

Jörg Apitzsch, bremen online services GmbH & Co.

ja@bos-bremen.de

Inhaltsverzeichnis

1 Vorbemerkungen

Die in diesem Dokument dargestellten Änderungen und Erweiterungen zur Spezifikation OSCI Transport 2.0 basieren primär auf Ergebnissen des Abstimmungsprozesses im Projekt XTA des IT-Planungsrates.[1]

Die OSCI-Spezifikationen adressieren bisher neben einem allgemeinen Rollenmodell der involvierten Kommunikationspartner und –knoten primär das Austauschprotokoll zwischen OSCI-Knoten. Die Gestaltung der Schnittstelle der OSCI-Knoten zu Fachanwendungen wurde offen gelassen, um Raum zu lassen für Implementierungen gemäß unterschiedlichen Anforderungen, differierenden Netzwerktopologien, jeweils genutzten Verzeichnisdiensten etc.

Allerdings stellt die Realisierung und Pflege der Schnittstelle zu Fachverfahren einen bedeutenden Kostenfaktor dar; deshalb besteht ein hoher Bedarf an ein vereinheitlichtes „Convenience-Interface“ zur Anbindung von Fach- an Transportverfahren, um so die Komplexität von OSCI-Transport zu kapseln. Das Projekt XTA des IT-Planungsrates hat unter anderem die Standardisierung dieser Schnittstelle zum Gegenstand.[2]

Ausgehend von den XTA-Projektergebnissen wird OSCI 2.01 in folgenden Bereichen weiterentwickelt:

Abbildung der Metainformationen

OSCI2.01 spezifiziert einen (optionalen) Metadaten-Container **`MessageMetaData`** als SOAP-Header. Dieser Header soll vollständig die XTA-Anforderungen bedienen, aber auch jene anderer Szenarien wie z.B. das BMI-Vorhaben P23R[3] oder die Cross-Solution/Cross-Border Lösungen wie in den *Large Scale Pilot* Projekten (LSP) der EU Kommission[4] konzipiert und pilotiert. Für Transportknoten (auch Knoten, die auf der Transportstrecke relativ zu einem OSCI-Gateway einem Author nachgelagert / Reader vorgelagert sind) ist kein Parsen des SOAP-Body (Payload) zur Erfüllung der jeweiligen Funktionalität erforderlich.

Dabei gibt es neben durchgängig in allen Szenarien genutzten Metainformationen auch solche, die nur innerhalb bestimmter Geschäftsbereiche relevant sind[5]. Ein solcher Ansatz greift u.a. auch Erfahrungen aus b2b-Anwendungen auf – für aktive Transportknoten sichtbare

[1] XTA: Entwicklung eines fachunabhängigen Standards für Transportverfahren; http://www.xoev.de/sixcms/detail.php?gsid=bremen83.c.4835.de

[2] XTA: Entwicklung eines fachunabhängigen Standards für Transportverfahren; http://www.xoev.de/sixcms/detail.php?gsid=bremen83.c.4835.de

[3] Details: http://www.p23r.de/

[4] https://ec.europa.eu/digital-agenda/en/egovernment-large-scale-pilot-projects

[5] Daraus ergibt sich die Anforderung, neben einem „Standard-Set“ ist auch die Möglichkeit von Profilierungen zusätzlicher Metainformation für spezifische Szenarien oder sogar Nachrichtentypen vorzusehen(z.B. XÖV).

Metainformationen dienen dem automatisierten „Handshake“ der Funktionalitäten auf Seiten von Service Requestor (OSCI-Rolle: Author) und Service Provider (OSCI-Rolle: Reader).

Eine Aufweichung der „minimalen“ Sichtbarkeit von Metainformationen (also der Beschränkung auf die Auszeichnung des zugeordneten Geschäftsszenarios oder Nachrichtentyps) scheint akzeptabel unter der Voraussetzung, dass aktive OSCI-Transportknoten vertrauenswürdige Dienste sind, deren Betrieb strengen Regelungen bzgl. Vertrauenswürdigkeit und Sicherheit unterliegen.[6]

Informationen im **`MessageMetaData`** werden teilweise auch durch Transportknoten ergänzt (z.B. Zeitpunkte). Die hier enthaltenen Informationen sollen auch in die Logs der Transportknoten aufgenommen werden (Nachvollziehbarkeit, Revision). In den Entwurf sind Erkenntnisse aus den LSPs und der OASIS Spezifikation ebXML Messaging Services (ebMS, Version 3.0)[7] eingeflossen.

Schrittweise Öffnung für generische Adressierung

OSCI2 als SOAP-Profilierung mit http-Binding und Adaption des w3c-Standards WS-Addressing benötigt http-Endpunkte für die technische Adressierung. Adressen in dieser Form sind für Fachanwendungen und/oder Nutzern von OSCI-Infrastrukturen nur schwer, zumindest nicht intuitiv handhabbar; sie werden i.d.R. auch heute schon aus generischen Adressen für Kommunikationsendpunkte gewonnen – so über Selektion von EGVP-Teilnehmern in S.A.F.E. oder AGS / Dienstebezeichnung im DVDV.

Im Kontext der Verbindung unterschiedlicher Zustelldienste in den EU-LSPs kommt erschwerend hinzu, dass eine Selektion von Adressaten in Verzeichnisdiensten fremder Infrastrukturen nicht möglich ist; auch existiert eine Vielzahl unterschiedlicher logischer unterschiedlicher technischer Adressformate.

Es bietet sich daher an, OSCI schrittweise zu öffnen für den Zugang mit generischen Adressen, wie sie von „User Agents“ (wie z.B. EGVP) bzw. auch Fachanwendungen gehandhabt werden. Eine Umsetzung in technische Adressen (WS-Addressing http-Ressourcen) könnte ein zusätzlicher Service eines OSCI-Gateways werden, bei dem über den Typ des generischen Identifiers eines Adressaten entschieden wird, über welchen Verzeichnisdienst in technische Adressen und weitere Routinginformationen aufgelöst werden kann.

[6] Hier besteht sicher noch rechtlicher Handlungsbedarf, vergleichsweise etwa zum De-Mail Gesetz oder den Anforderungen des oben erwähnten Entwurfs der EU-Regulierung: Dienstleister für OSCI-Transportinfrastrukturen (Betreiber von Intermediären, aber auch zugeordneten Verzeichnisdiensten) müssen Konformitätsregelungen (sichere, vertrauenswürdiger Betrieb) und entsprechender rechtlicher Verantwortlichkeit und Kontrolle unterliegen. Nutzer müssen sich auf die zu regulierende Konformität der Betreiber verlassen können; die Konformität der eingesetzten Systeme zu Spezifikationen wie hier hier OSCI Transport mit definierten Sicherheitsanforderungen ist i.d.R. bereits durch die Produkthaftung der Hersteller geregelt.

[7] http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms_core-3.0-spec.pdf

Erhöhte Flexibilität für die Umsetzung von Einsatzszenarien

Die Spezifikation OSCI Transport 2.0 definiert eine Reihe obligatorischer Anforderungen für konforme Implementierungen, die zum Teil in Einsatzszenarien nicht benötigt werden; andererseits bestehen Anforderungen an zusätzliche Funktionalitäten, die in die Spezifikation Version 2.01 aufgenommen werden.

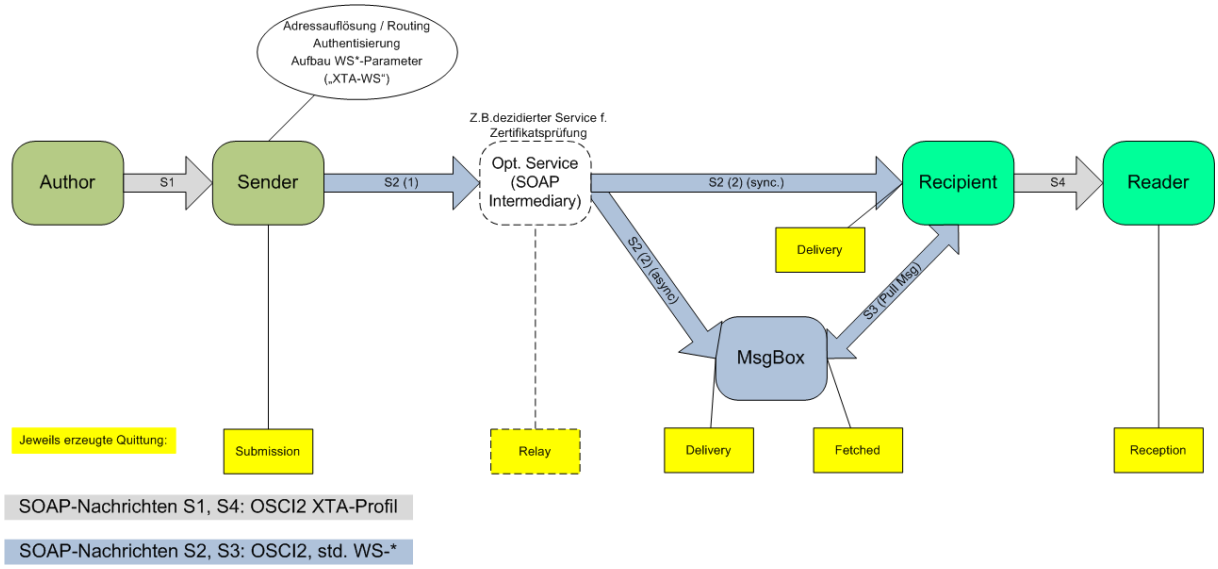
Insgesamt werden mit der Version 2.01 Änderungen und Erweiterungen der Version 2.0 vorgenommen und in Implementierungen verfügbar gemacht werden, die erhöhte Flexibilität bzgl. der Einsatzszenarien und Anforderungen aus den oben erwähnten Projekten adressieren.

Eine Implementierung von OSCI 2.01 ist kurzfristig realisierbar und sichert weitestgehend Abwärtskompatibilität zur Version 2.0.

2 Benutzte Namenskonventionen, OSCI-Rollen

Mit OSCI V2.01 wird eine dezidierte „Sender“-Rolle eingeführt, die funktional dem XTA-WebService, Port SendPort entspricht. Die „Sender“-Rolle als dezidierter Knoten war im Rahmen der Arbeiten an der OSCI2-Spezifikation schon einmal als „Dispatcher“ modelliert worden, letztlich aber samt „Dispatcher“-Knoten verworfen worden. Entsprechend erweitert sind auch zusätzliche Quittungen vorzusehen.

Grobe Veranschaulichung:



In Version 2.01 ist zunächst die zusätzliche „Submission“-Quittung vorgesehen; aktuell existiert kein Szenario mit weiteren optionalen Services auf dezidierten aktiven Knoten der Transportstrecke.

Die unterschiedlichen Web Services Spezifikationen folgen leider nicht durchgängig einer einheitlichen Nomenklatur mit klar definierter Semantik der jeweiligen Begriffe.

Zwecks Klarheit werden hier Nomenklatur und Semantik der OSCI-Welt noch einmal mit jenen abgeglichen, wie sie die w3c „Web Services Architecture Working Group“ 2004 beschrieben hat in „Web Services Glossary“[8] und „Web Servcies Architecture“[9] (WS-Arch).

- **Author - Requester**
  „The requester entity is the person or organization that wishes to use a provider entity's Web service.… The requester agent is the software agent that requires a certain function to be performed on behalf of its owner — the requester entity. From an architectural perspective, this is the agent that is looking for and invoking or initiating an interaction with a provider agent” (WS-Arch).

[8] http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/#defs

[9] http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

„Autor“ im Sinne des OSCI-Rollenmodells; auch Fachverfahren, Initiator Source Application. Quelle des Payload (s.u.) und initialer Attribute zum Transportauftrag.

- **Sender - Sender**
  „A message sender is the agent that transmits a message“ (WS-Arch). „Sender“ im Sinne des OSCI-Rollenmodells; implementiert ein OSCI Gateway, evt. mit vorgelagerten Funktionen (wie z.B. ein XTA-WebService). Auch „Dispatcher“: Dienst, dem sich ein Requester / Author zur Nachrichtenübermittlung bedient.

- **Recipient - Receiver**
  „A message receiver is an agent that receives a message“ (WS-Arch). „Empfänger“ im Sinne des OSCI-Rollenmodells; implementiert ein OSCI Gateway, evt. mit nachgelagerten Funktionen (wiederum wie z.B. ein XTA-WebService, der einem Provider Agent (s.u.) Funktionen zum Zugriff auf einen MsgBox-Knoten zur Verfügung stellt).

- **MsgBox –** (in etwa) **Proxy**
  „Intermediär“ im Sinne des OSCI1.2-Rollenmodells; im OSCI2-Modell Spezialfall eines Receiver für asynchrone Zustellung (Message Relay Funktionalität). W3C Glossary: “An agent that relays a message between a requester agent and a provider agent, …”

- **Reader - Provider**
  „Leser“ im Sinne des OSCI-Rollenmodells; „Target Application“; in OSCI2 (in Anlehnung an SOAP) auch „Ultimate Recipient“.

- **Inhaltsdaten - Payload**
  Fachliche Inhaltsdaten, die zwischen Author und Reader (Requester und Provider) übermittelt werden. Transparent für alle Transportknoten.

- **Transportauftrag - Message Meta Data (Header)**
  Transportdaten (XÖV: auch „Transportauftrag“) und Metainformation zum Payload, die für Transportknoten sichtbar müssen, ggf. auch erst für eine dem Recipient-Knoten nachgelagerte Routing- und Selektions-Funktionalität vor Verarbeitung des Payload durch den Knoten Reader. WS-Architecture: „The message envelope may contain information needed to actually deliver messages. If so, it must at least contain sufficient address information so that the message transport can deliver the message. …. Although many headers will relate to infrastructure facilities, such as security, routing, load balancing and so on; it is also possible that headers will be application specific. …

  The primary function of headers is to facilitate the modular processing of the message, although they can also be used to support routing and related aspects of message processing. The header part of a message can include information pertinent to extended Web services functionality, such as security, transaction context, orchestration information, message routing information, or management information.

  Message headers may be processed independently of the message body, each message header may have an identifying service role that indicates the kind of processing that should be performed on messages with that header. Each message may have several headers, each potentially identifying a different service role.”

- **Intermediär - Intermediary**
  WS-Glossay: “A service intermediary is a Web service whose main role is to transform messages in a value-added way. (From a messaging point of view, an intermediary processes messages en route from one agent to another.) Specifically, we say that a service intermediary is a service whose outgoing messages are equivalent to its incoming messages.”

  Eine OSCI MsgBox ist ein Intermediär mit der speziellen Service-Funktionalität der sicheren Zwischenspeicherung von Nachrichten und (SOAP-) Pull-Funktionen zum Abholen dieser durch authentisierte Recipients.

3 Änderungen / Erweiterungen zu V 2.0 in V 2.01

Mit der Version 2.01 sollen kurzfristig zunächst Änderungen und Erweiterungen der Version 2.0 vorgenommen und in Implementierungen verfügbar gemacht werden, die erhöhte Flexibilität bzgl. der Einsatzszenarien und Anforderungen aus dem XTA-Projekt adressieren. Die hier dargestellten Punkte sind kurzfristig realisierbar und sichern weitestgehend Abwärtskompatibilität zur Version 2.0.

3.1 Unterstützung logischer Adressierung

OSCI2 als SOAP-Profilierung mit http-Binding und Adaption des w3c-Standards WS-Addressing benötigt http-Endpunkte für die technische Adressierung. Adressen in dieser Form sind für Fachanwendungen und/oder Nutzern von OSCI-Infrastrukturen nur schwer, zumindest nicht intuitiv handhabbar; sie werden i.d.R. auch heute schon aus generischen Adressen für Kommunikationsendpunkte gewonnen – so über Selektion von EGVP-Teilnehmern in S.A.F.E. oder AGS / Dienstebezeichnung im DVDV.

Im Kontext der Verbindung unterschiedlicher Zustelldienste in den EU-LSPs kam erschwerend hinzu, dass eine Selektion von Adressaten in Verzeichnisdiensten fremder Infrastrukturen nicht möglich ist; auch existiert eine Vielzahl unterschiedlicher logischer unter technischer Adressformate.

Es bietet sich daher an, OSCI schrittweise zu öffnen für den Zugang mit generischen Adressen, wie sie von „User Agents“ (wie z.B. EGVP) bzw. auch Fachanwendungen gehandhabt werden. Eine Umsetzung in technische Adressen (WS-Addressing http-Ressourcen) könnte ein zusätzlicher Service eines OSCI-Gateways werden, bei dem über den Typ des generischen Identifiers eines Adressaten entschieden wird, über welchen Verzeichnisdienst in technische Adressen und weiter Routinginformationen aufgelöst werden kann.

Kommunikationsendpunkte werden durch einen Typ „**`PartyIdentifierType`**“ modelliert. Der Identifier selbst ist vom Typ **`xs:normalizedString`**, attributiert durch **`@type`**, vom Type **`xs:QName,`** welches den Typ dieses Identifiers auszeichnet – also z.B. einen Identifier aus dem DVDV, aus S.A.F.E. oder – im Falle spezieller P23R-Bedarfe – ein noch festzulegender Qname, der **`TBRS:notificationProfile/communication/receivers/@receiverId`**[10] repräsentiert. Aus dem jeweils zugeordneten Verzeichnisdienst können die Verbindungsparameter entnommen werden, wie sie OSCI für WS-Addressing benötigt (letztlich Lokation Recipient, MsgBox).

Ein **`PartyIdentifierType`** hat zusätzliche optionale informatorische Attribute, die hier bereits aufgrund der Anforderungen aus den EU LSPs mit Anlehnung an den entsprechenden Typ „PartyIdentifier“ aus der ebXML Messaging Service Spezifikation[11] aufgenommen wurden.

[10] Zu verifizieren durch Projekt P23R

[11] http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/ebms_core-3.0-spec.pdf

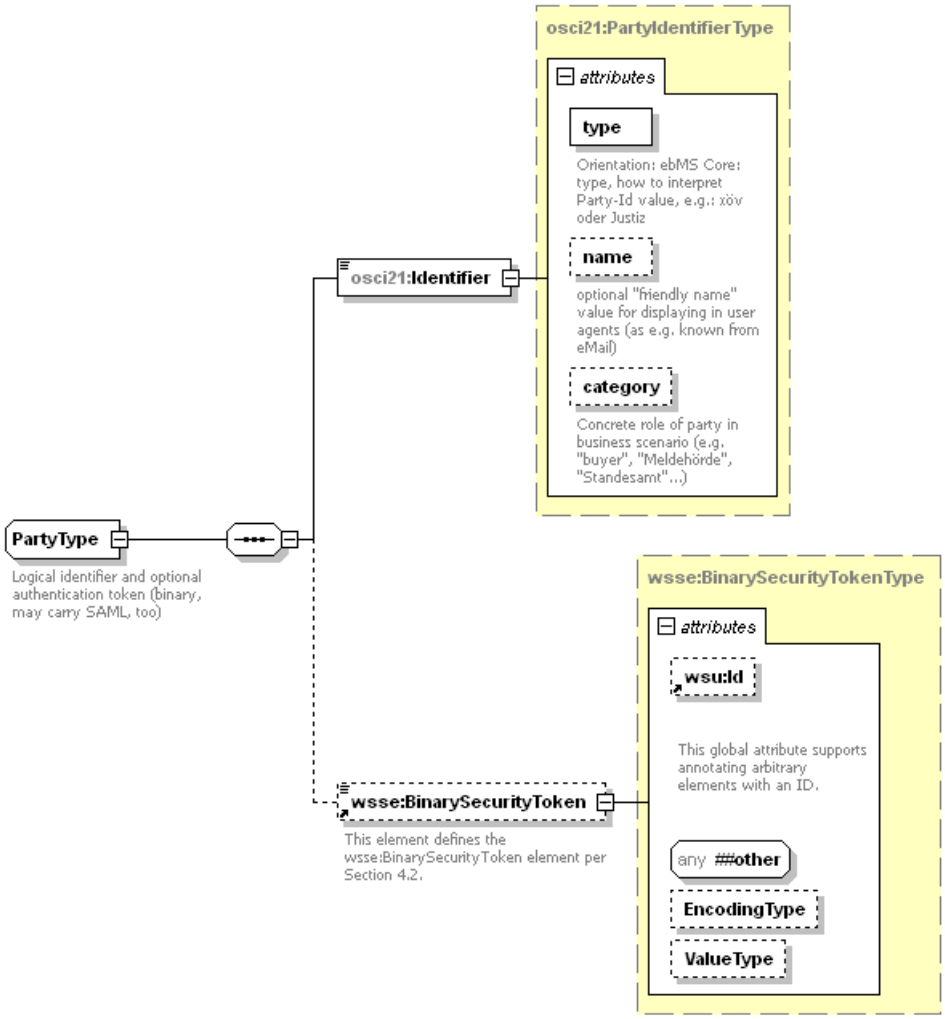
```
<xs:complexType name="PartyIdentifierType">
    <xs:annotation>
      <xs:documentation>Value of generic party identifier, as classified by
@type attribute,  e.g.: Prefix:Kennung</xs:documentation>
    </xs:annotation>
    <xs:simpleContent>
      <xs:extension base="xs:normalizedString">
        <xs:attribute name="type" type="xs:QName" use="required">
          <xs:annotation>
            <xs:documentation>Orientation: ebMS Core: type, how to interpret
Party-Id value, e.g.: xöv oder Justiz</xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="name" type="osci21:NonEmptyStringType">
          <xs:annotation>
            <xs:documentation>optional "friendly name" value for displaying
in user agents (as e.g. known from eMail)</xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="category" type="xs:QName">
          <xs:annotation>
            <xs:documentation>Concrete role of party in business scenario
(e.g. "buyer", "Meldehörde", "Standesamt"...)</xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
</xs:complexType>
```

PartyIdentifierType ist die Typdefinition für die Instanzen der Source- und Target-Endpukte **Originators (**Author, auch Sender) und **Destinations** (Recipient, andere Empfänger).

Eine entsprechende Auflösung generischer Adressen ist erst für eine Folgeversion von OSCI 2.01 geplant; zunächst sind die WS-Addressing Parameter noch von einer im Knoten Sender, dem OSCI-Gateway vorgelagerter Funktionalität zu setzen (wie z.B. vom XTA WebService).

In vielen Szenarien besteht die Anforderung, zu einem Kommunikationsendpunkt (oder auch Knoten) Authentisierungsinformationen zu übermitteln; i.d.R. sind dies X509- oder SAML-Token.

Für diese Zwecke wird folgender Typ **PartyType** definiert, der einen **PartyIdentifier-Type** um ein optionales Element **BinarySecurityToken** gem. WS-Security erweitert:

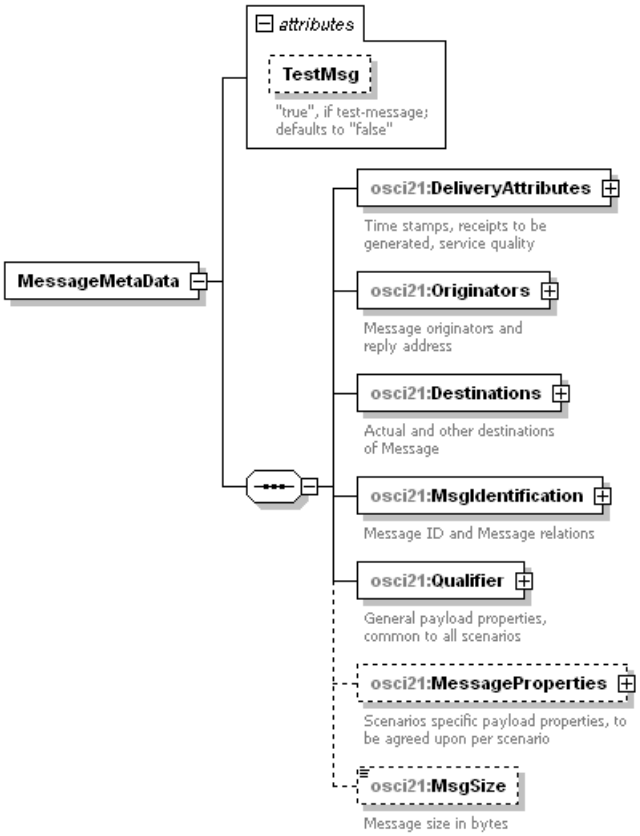


Die dargestellten komplexen Typen zur Aufnahme generischer Adressierungsinformationen werden instanziiert in den Elementen, die in einem neu definierten Header **`MessageMeta-Data`** enthalten sind.

3.2 Neuer Header MessageMetaData

Mit Blick auf eine längerfristig geplante Version 2.1 der Spezifikation ist der dieser Header ist bereits im Namensraum **`"http://www.osci.eu/ws/2013/02/transport"`** definiert (Präfix osci21:).

3.2.1 Übersicht

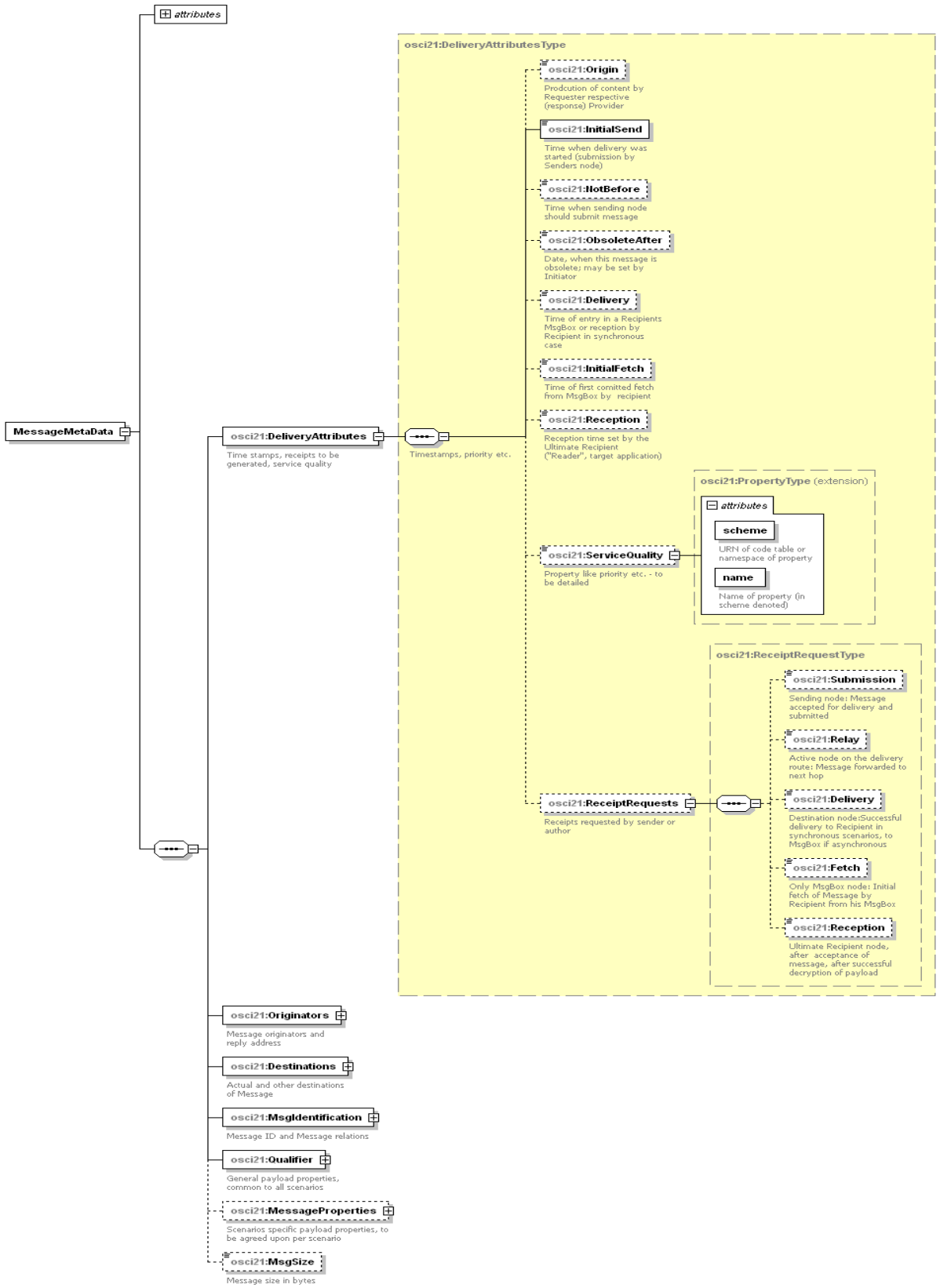


Der Header-Block **`MessageMetaData`** nimmt alle Transportdaten und Metainformationen zum Payload auf. Er ist attributiert durch **`xs:boolean @TestMsg`**, um bei Bedarf Testnachrichten auszuzeichnen.

`MessageMetaData` hat generelle Bestandteile, die für alle unterstützten Szenarien relevant sind. Weiter können im optionalen Element **`MessageProperties`** Szenarien spezifische Meta-Informationen zum Payload hinterlegt werden, die entsprechend pro Szenario profiliert werden müssen („Property"-Benennung, -Werte, -Semantik). Element **`MsgSize`** dient lediglich der Optimierung bei Empfangsknoten (Streaming) und wird vom sendenden OSCI-Gateway mit der Größe der Nachricht in Bytes gesetzt.

3.2.2 Generelle Transportdaten

3.2.2.1 DeliveryAttributes



- Zeitstempel, die von entsprechenden Knoten während des Transports appliziert werden (siehe auch **MsgTimeStamps** auch OSCI2.0)
  - **Origin, ObsoleteAfter** können vom Author gesetzt werden
  - **InitialSend** muss vom Sender gesetzt werden
  - **NotBefore** kann vom Autor gesetzt werden; Sender muss in diesem Fall die Nachricht erst zu diesem Zeitpunkt versenden
  - **Delivery** muss von MsgBox (asyn.) bzw. Recipient (syn.) gesetzt werden
  - **InitialFetch** muss von MsgBox beim initialen Abholen einer Nachricht gesetzt werden
  - **Reception** muss vom Reader gesetzt (bzw. durch diesen Knoten getriggert) werden

  Der in OSCI2.0 definierte Header **MsgTimeStamps** wird aus Kompatibilitätsgründen weiter in OSCI-Nachrichten mitgeführt. Die Einträge in **MsgTimeStamps** müssen von OSCI2.01-Gateways bzw. OSCI2.01-MsgBox-Services entsprechend der oben aufgeführten Elemente in **MessageMetaData** transparent gesetzt werden. Der damit redundante Header **MsgTimeStamps** kann daher in einer zukünftigen Version von OSCI Transport entfallen.

- **ServiceQuality**: Prioritäten und andere Eigenschaften, noch zu detaillieren. Der Typ „**PropertyType**“ ist eine Extension von **xs:anySimpleType** mit qualifizierende Attributen **@name** und seinem Namensraum **(@scheme**) dieser Property:

```
<xs:complexType name="PropertyType">
  <xs:simpleContent>
    <xs:extension base="xs:anySimpleType">
      <xs:attribute name="scheme" type="xs:anyURI" use="required">
        <xs:annotation>
          <xs:documentation>URN of code table or namespace of prop-
erty </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="name" type="xs:QName" use="required">
        <xs:annotation>
          <xs:documentation>Name of property (in scheme denot-
ed)</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

- **ReceiptRequests**: Quittungsanforderungen als Sequenz von leeren Elementen (beim Knoten Sender konfigurierbar gem. Szenario oder explizit

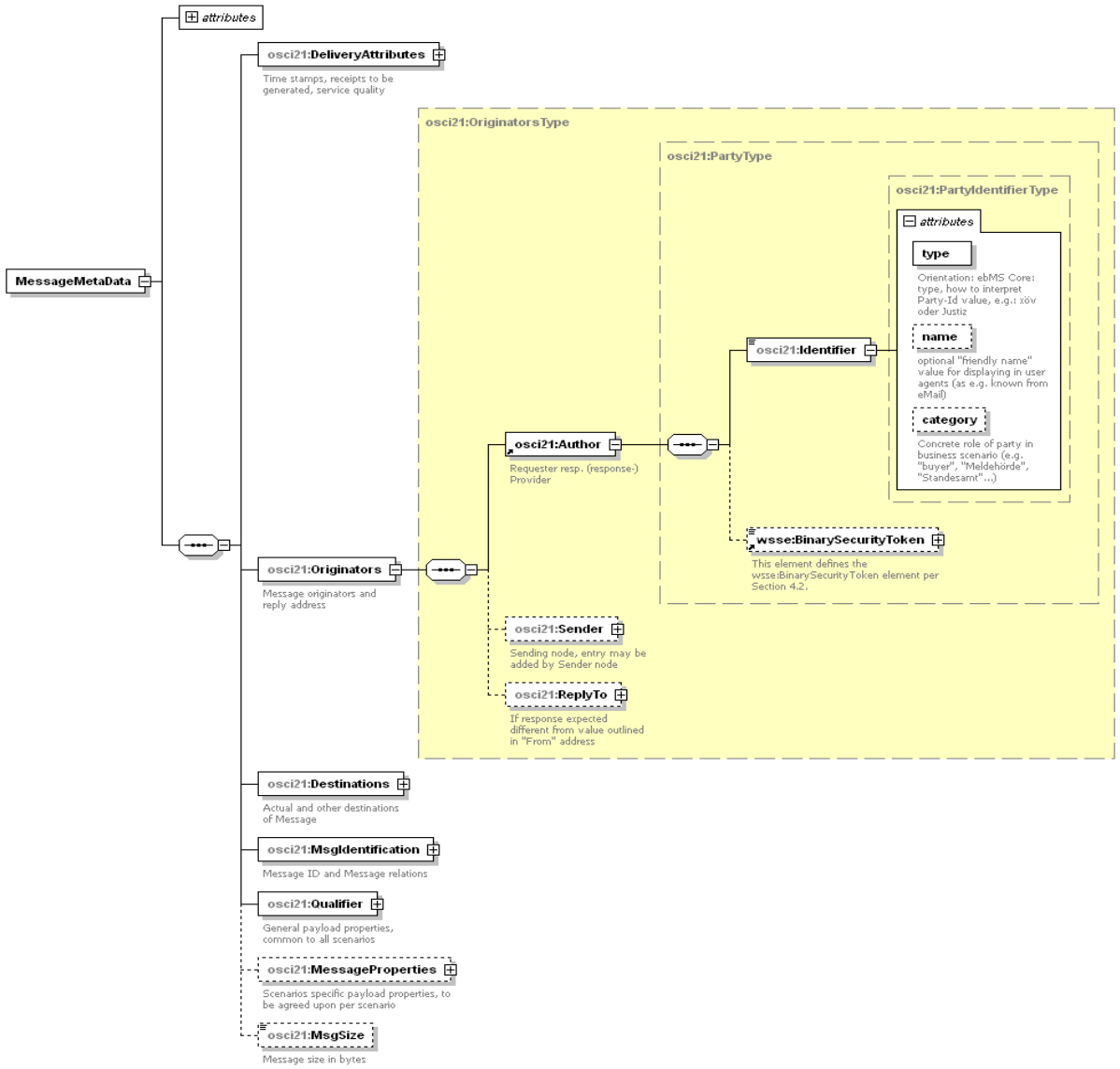
durch Initiator gesetzt – dann Vorrang vor Konfiguration); wird vom sendenden OSCI-Gateway in die in OSCI 2.0 definierten Receipt-/Notification Requests umgesetzt.[12]

Für die Parametrierung der Receipt-/Notification Requests gelten in diesen Fall Default-Werte gem. folgender Überlegungen: Die „ReceiptRequests“ von OSCI2.0 scheinen „oversized“. Quittungen sollten immer zum Sender ausgeliefert werden, ein „Echo“ des übermittelten Payload wird offenbar nicht benötig (Sender bzw. Author sollten bei absehbaren späteren Nachweisbedarfen Kopien Nachrichten aufbewahren). Bzgl. der Anforderung qual. Zeitstempel für Quittung sind bisher keine realen Bedarfe sichtbar geworden. Es bleibt abzuwarten, ob hier in Zukunft die eIAS-Regulierung der Kommission greift, die im Entwurf generell qualifizierte Zeitstempel für Quittungen fordert.

Ist in **`ReceiptRequests`** auch „Submission“ gesetzt, erzeugt der Knoten „Sender“ eine (neu definierte) Quittung „**`SubmissionReceipt`**“, Format analog „**`DeliveryReceipt`**“.

[12] Diese in OSCI 2.0 definierten SOAP-Header werden aus Gründen der Abwärtskompatibilität weiterhin aufgebaut. In einer weiteren zukünftigen Version könnte auf diese Header verzichtet werden.

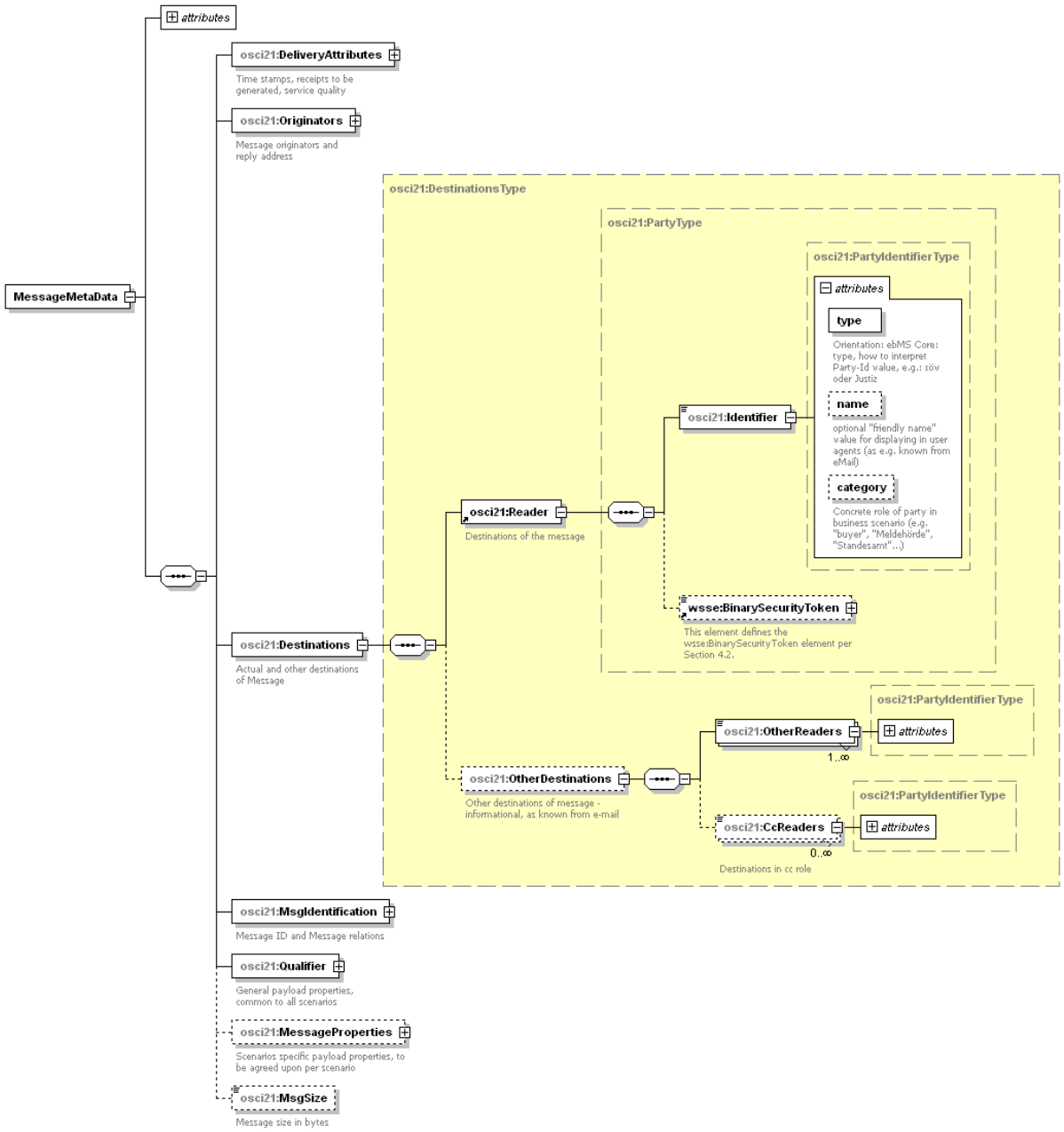
3.2.2.2 Originators



Alle Elemente sind vom Typ **`PartyType`** – Details siehe Kapitel 3.1.1. Neben dem generischen Identifier (**`PartyIdentifierType)`** können Authentisierungstoken (X509, auch SAML) aufgenommen werden.

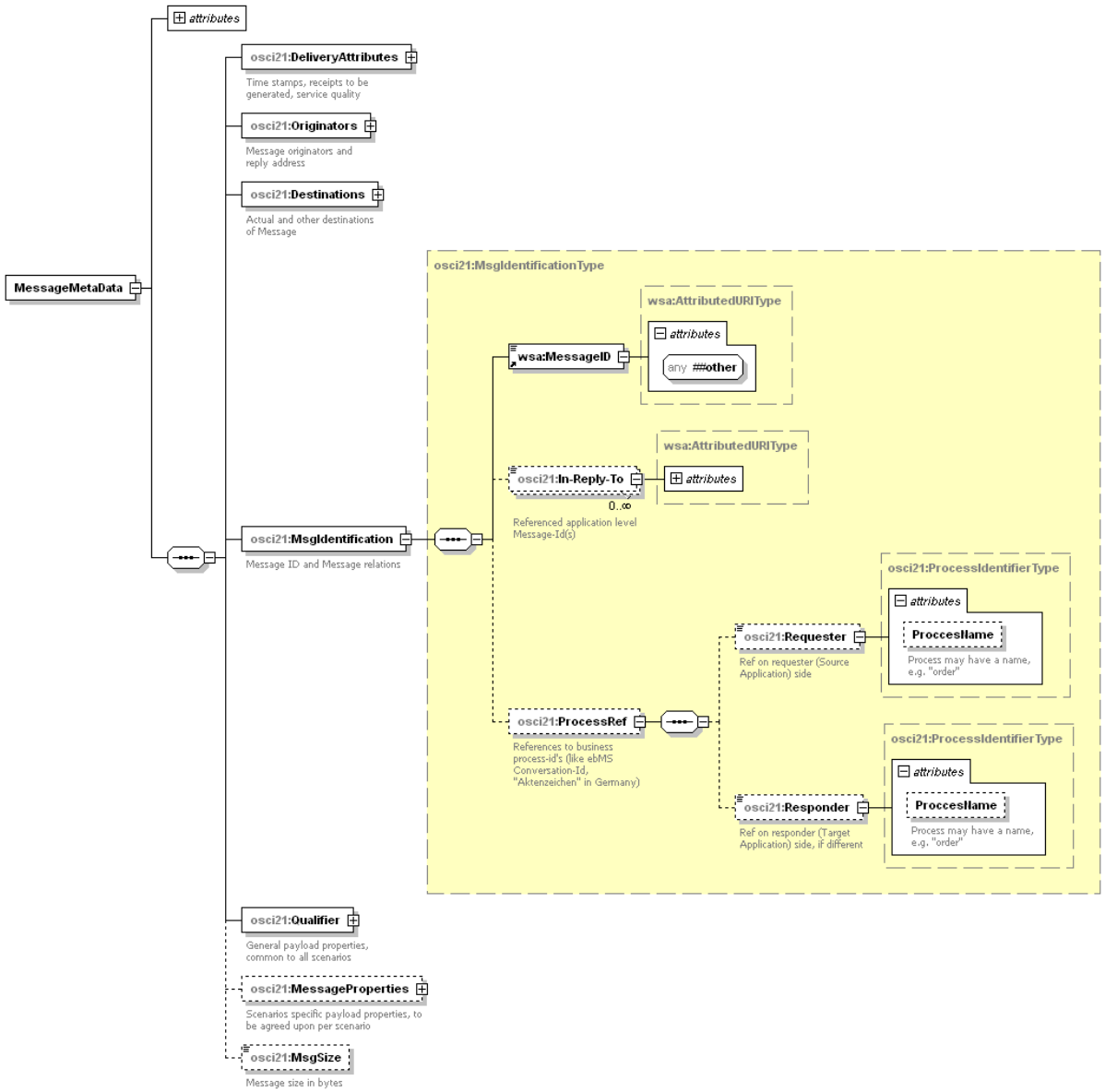
- **`Author`** – obligatorisch entspr. Initiator, Source Application, WS Addressing **`From`**)
- **`Sender`** (sendender Knoten/OSCI Gateway, z.B. XTA-WS)
- **`ReplyTo`** – Zieladresse, an die Antwort gesendet werden soll; default = **`Author,`** entsprechend Semantik von WS-Addressing.

3.2.2.3 Destinations



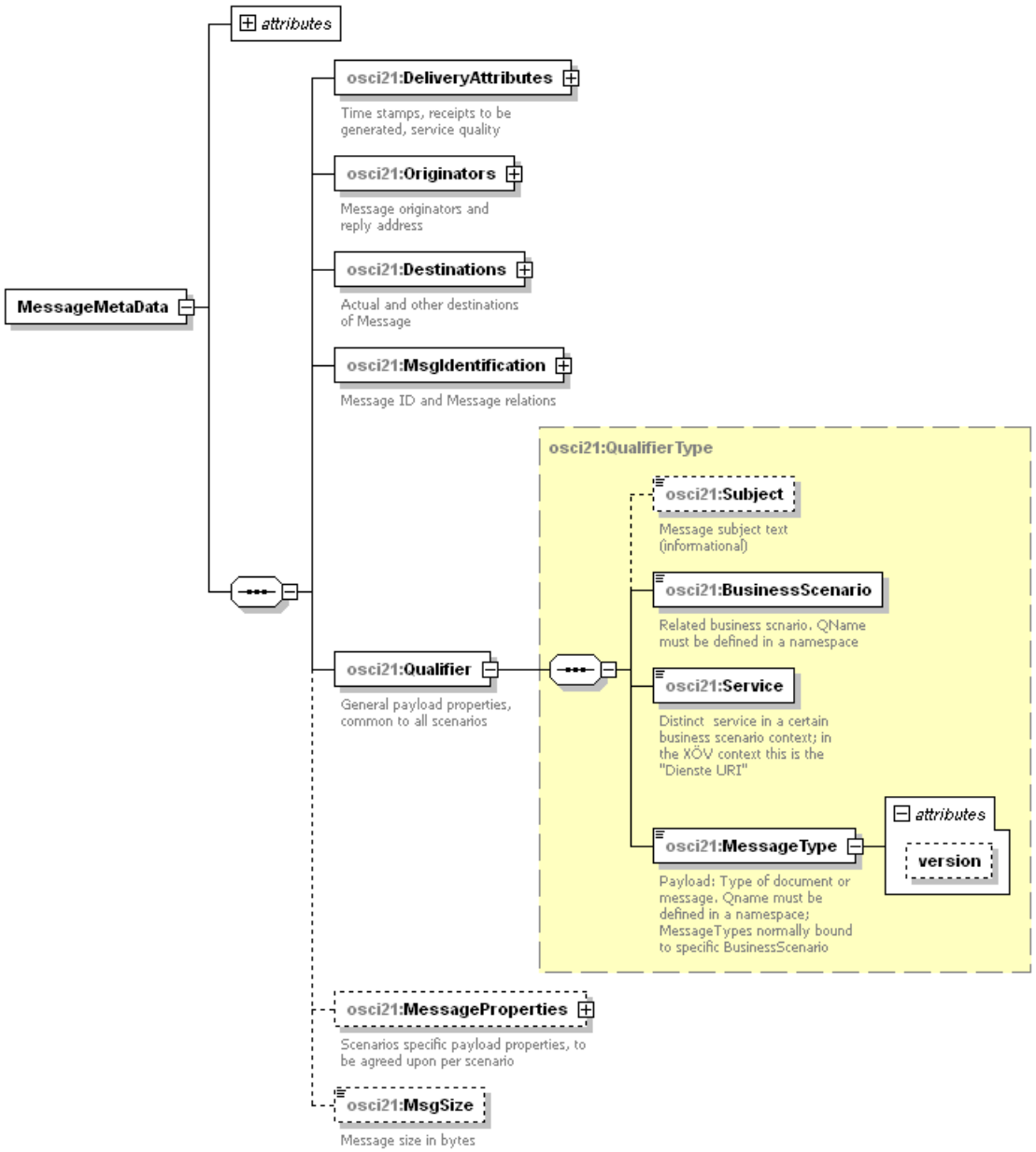
- **`ReaderIdentifier`** – obligatorisch; entspricht Target Application, Service Provider. Muss vom Author gesetzt werden
- **`OtherRecipients -`** informatorisch, optional: weitere Adressaten der Nachricht, unterschiedliche in Rollen to: und cc:; hier ist nicht vorgesehen, zum generischen „**`PartyIdentifer`**“ auch optionale Authentisierungstoken aufnehmen.

3.2.2.4 MsgIdentification



- **`Message-Id`** – obligatorische Id der Nachricht, wie vom Author (Applikationsebene) vergeben
- **`In-Reply-To -`** bezogene Nachricht(en) (Applikationsebene):
- **`ProcessRef -`** Bezug zu laufenden Vorgang (z.B. Aktenzeichen)
    - Unterscheidung zwischen Vorgangsnummer auf Requester- und Responder-Seite möglich
    - Vorgangsnummer selbst kann mit einem Vorgangsnamen **`ProcessName`** attributiert werden

3.2.2.5 Qualifier

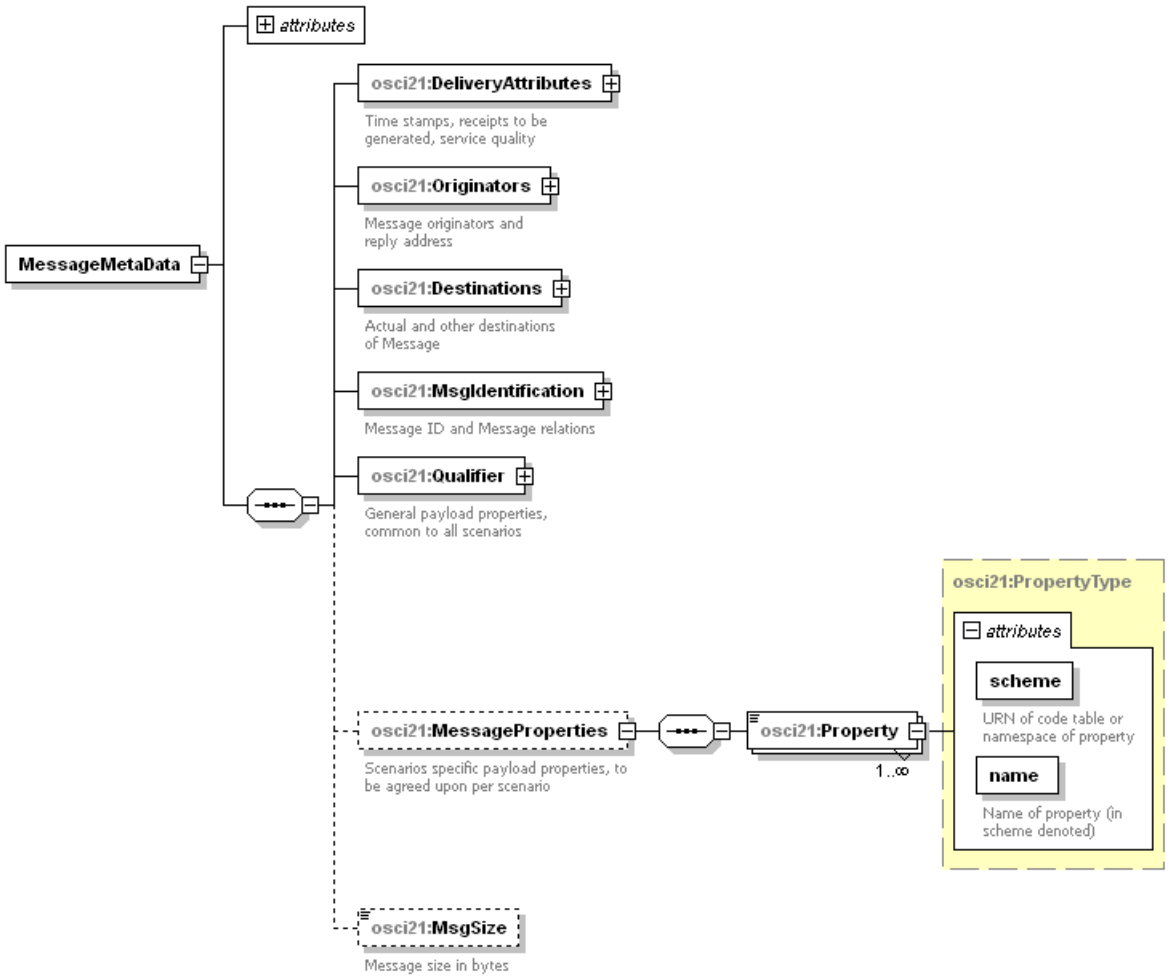


- **`Subject`** – rein informatorischer Begleittext (aka E-Mail „about“)
- **`BusinessScenario`** – obligatorisch, Geschäftsszenario als **`xs:QName`** (ersetzt das OSCI2.0 „BusinessScenarioType); QNames sind festzulegen!
- **`Service`** – obligatorisch, Dienst des “Reader” (Target Application) als URI **`Message-Type`** (**`xs:anyURI`**)– obligatorisch Nachrichtentyp innerhalb des Geschäftsszenarios als **`xs:QName`**; bindet ein definiertes Payload-Schema; wie im Meldewesen gebräuchlich ggf. attributiert durch eine **`@version`**

3.2.3 Szenarien spezifische Metadaten zum Payload

3.2.3.1 MessageProperties

Dieses Element ist vorgesehen, um bei Bedarf Metainformationen aufzunehmen, die Szenarien bezogen benötigt werden. Semantik und Verarbeitung dieser Informationen sind außerhalb des Fokus von XTA und OSCI-Transport.



`MessageProperties`

- Dieser Container nimmt **`Property`**-Elemente auf, die innerhalb eines definierten Geschäftsszenarios festzulegen sind (Name, Werte, Semantik).

`Property`

- Im Kern ist eine Property ein Key/Value-Pärchen innerhalb eines bestimmten Namensraums. Für Kommunikationsszenarien innerhalb eines definierten Geschäftsszenarios müssen entsprechende Listen festgelegt werden, um identische semantische Interpretation sicher zu stellen. Das Element ist vom Type **`xs:anySimpleType`** und kann damit z.B. auch base64-codierte Werte aufnehmen, wie z.B. von XTA adressiert.

`Property,` @scheme

- Das Attribut **`@scheme`** definiert den Namensraum einer Property (z.B. Geschäftsszenario).
- XTA bezieht sich für Payload-Auszeichnung u.a. auf XÖV-Codetabellen, was nicht unbedingt für alle von OSCI zu unterstützenden Szenarien handhabbar ist. Der Bezug zu solchen „Properties“ kann aber durch eine entsprechende URN als Wert im Attribut **`@scheme`** von **`Property`** abgebildet werden.

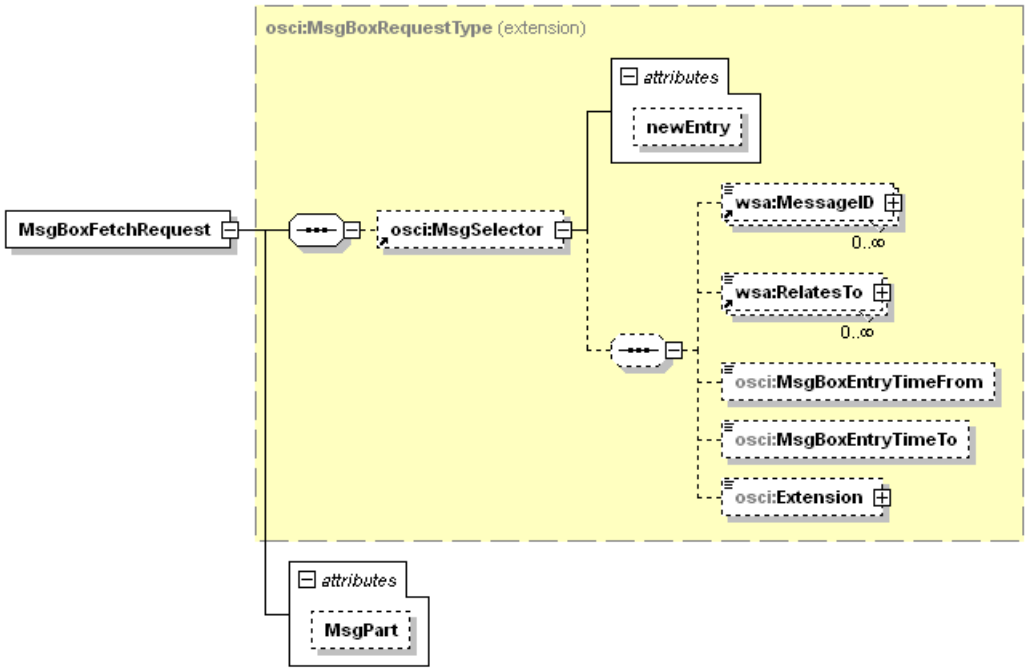
3.3 Zugriffe auf die OSCI MsgBox

- Die diversen obligatorischen Selektionsmöglichkeiten der Pull-Mechanismen auf die OSCI Message-Box **`(MsgSelector)`** werden z.T. nicht genutzt. Oft reicht der Zugriff über eine dezidierte MessageId oder den Status (nur neu eingegangene Nachrichten). Implementierungen sollte anheimgestellt werden, nur die jeweils benötigen Mechanismen zu implementieren.
- Für spezifische zusätzliche Zugriffsmöglichkeiten soll eine Selektion über flexible Mechanismen gem. Anwendungsanforderung zugelassen sein, die sich auf alle Elemente und Attribute aus Kapitel 1.3.2. (Metadaten und 1.3.3 (logische Adressen, als Teil eine neuen Metadaten-Headers vorgesehen, s.u.) beziehen. Technisch ist dies abbildbar durch XPath-Ausdrücke, wie in der Spezifikation OSCI 2.0 in Abschnitt 8.2 (**`MsgSelector/Extension/{any} *)`** bereits prinzipiell vorgesehen und muss in Bezug auf den neuen Metadaten-Header konkretisiert werden.
- Bisher ist für den Body eine **`MsgBoxResponse`** spezifiziert, dass hier die gesamte Ursprungsnachricht inklusive der Transport-Header bereitgestellt wird. Erfahrungen haben gezeigt, dass die Gesamtheit dieser Informationen i.d.R. nicht genutzt werden, zusätzlich erhöht dieses Konzept dem Implementierungsaufwand. Um den Zugriff vor allem auch auf Seite des Clients (Software den Empfängers) zu vereinfachen (reduziertes Parsen) wird angeregt, dies zu ersetzen durch folgende Container:
    - Den Body (Inhaltsdaten) der zugestellten Nachrichten
    - den neu definierten Metadaten-Container
    - Security Token (Transport!) des WS-Security Headers
    - Message Time Stamps

    Weiter, falls in Ursprungsnachricht vorhanden:

    - Header-Elemente **`X509TokenContainer`**, **`xkms:CompoundResult`**
    - **`ReceptionReceiptDemand`**

  Welche Informationen bereitgestellt werden sollen, wird durch ein neu definiertes Attribut **`@MsgPart`** in **`MsgBoxFetchRequest`** festgelegt:



@MsgPart ist vom Typ **xs:NMTOKEN** mit folgenden möglichen Ausprägungen:

- **Envelope** - stellt alle oben genannten Informationen (Header- und Body-Blöcke der ursprünglichen Nachricht) als **s12:Envolope** Elements im Body der **MsgBox-Response** bereit
- **Body** – stellt nur den Body der ursprünglichen Nachricht im Body der **MsgBox-Response** innerhalb eines **s12:Body** Elements bereit
- **Header** - stellt nur die oben genannten Header-Blöcke der ursprünglichen Nachricht im Body der **MsgBoxResponse** innerhalb eines **s12:Header** Elements bereit

3.4 SOAP Faults

OSCI 2.0 spezifiziert eine Liste vordefinierter OSCI SOAP Faults. Bereits bei der Verabschiedung der Spezifikation bestand die Erwartung, dass Erfahrungen aus Implementierung und Betrieb weitere Fehlersituationen sichtbar machen werden, die eine Erweiterung dieser Liste erforderlich machen wird. Inzwischen wird angeregt, dass Implementierungen bei Bedarf eigene SOAP Faults definieren können, um spezifische Fehlersituationen abzubilden.

3.5 wsa:Action

OSCI 2.0 spezifiziert eine Liste fest vordefinierter Werte für **wsa:Action**. In synchronen Szenarien erschwert dies die Implementierung, da aus dem Wert von **wsa:Action** die aufzurufende Funktionen des spezifischen Web-Services abgeleitet wird. Diese Aufzählung sollte entsprechend um eine Wert **xs:anyUri** erweitert werden; der Wert **http://www.osci.eu/ws/2008/05/transport/urn/messageTypes/OSCIRequest** wäre dann nur noch für die Zustellung in eine MsgBox anzuwenden. Um diese Semantik besser auszudrücken, kann in OSCI 2.01 auch der Wert

`http://www.osci.eu/ws/2013/02/transport/urn/messageTypes/MsgBoxInfeed` verwendet werden.

3.6 Aktualisierung der kryptografischen Algorithmen

Die Signatur- und Verschlüsselungsalgorithmen werden an den aktuellen Algorithmen Katalog der Bundesnetzagentur angepasst, der als Entwurf für das Jahr 2013 vorliegt und bis Ende 2019 geeignete Schlüssellängen und Algorithmen aufführt.

Dabei ist deutlich hervorzuheben, dass OSCI Transport 2.01 – wie schon OSCI 2.0 – primär Transportsignatur und -verschlüsselung adressiert. Die Vorgaben und Empfehlungen der Bundesnetzagentur beziehen sich auf die Anforderungen von SigG und SigV.

3.7 Generelles Review auf optionale und obligatorische Festlegungen

Im Sinne der vorgehend aufgeführten Punkte wird die Spezifikation insgesamt einer Bewertung optionaler und obligatorischer Aussagen unterzogen.