OSCI Transport 1.2

– Specification –

Status: FINAL

OSCI Leitstelle

Bremen/Germany, June 2002

The following companies and institutions have been involved in the creation of this specification:

- Bremen Online Services GmbH & Co. KG, Bremen/Germany
- cit GmbH, Dettingen unter Teck/Germany (MediaKomm Esslingen)
- OSCI Leitstelle, Bremen/Germany (editor)

  Am Fallturm 9
  D-28359 Bremen
  Internet: http://www.osci.de
- PPI Financial Systems GmbH, Hamburg/Germany
- SAP AG, Walldorf/Germany
- Stadtverwaltung Hagen (Council Administration of the City of Hagen/Germany)

The creation of this document within the MEDIA@komm-project was promoted by the Bundesministerium für Wirtschaft (the German Federal Department of Economic Affairs).

Copyright

Mark-Up Conventions

In this specification the following conventions are obeyed:

- Normative paragraphs are highlighted in light grey. Example:

  This paragraph is normative.

  In cases of doubt the fixings made in the schemas of this specification have priority to normative text passages of this specification. The latter ones in turn have priority to normative parts of documents referred to and these ones have priority to non-normative parts of this specification.
- Emphasized text is printed in *italics*.
- Any kind of code is printed in `typescript`.

Table of Contents

1. Introduction

OSCI Transport is a standard that in principle is able to automatically transfer any kind of information. The information can be signed (according to the German digital signature law) and protected against spying by means of encryption. Apart from that, OSCI Transport offers further security mechanisms concerning especially the traceability of the communication.

OSCI Transport forms part A of OSCI (Online Service Computer Interface), a standard for the area of e-government, which is currently being developed in a process of extended discussions and cooperation with the federal government, the states and local communities under the leadership of the OSCI Leitstelle in Bremen. Part A deals with the level of transport and security functions. Part B deals with the level of content/subject.

Essential design criteria for the present version of OSCI Transport were the following (cf. [OSCI-ANF]):

- Building on open standards

  Especially the following standards from the surroundings of web-services form the basis of OSCI Transport:
  - SOAP [SOAP 1]
  - XML Signature [XDSIG]
  - XML Encryption [XENC]
- Technology independence

  OSCI messages can be transmitted by any technical communication protocol.

  Furthermore, OSCI Transport does not make any specific demands regarding the platforms software systems processing OSCI messages are applied on or the programming languages these systems are developed in.
- Scalability of the security levels

  For example, to support simple business transactions advanced signatures may be used, whereas for transactions with the requirement of writing qualified or accredited electronic signatures can be applied.

This specification is addressed to software developers. To understand it, it is required to have knowledge of XML [XML], namespaces [XNAMESP] and XML Schema [XSCHEMA].

To keep this specification as concise and clear as possible, relevant standards are only referred to (and not paraphrased). The normative and non-normative parts of this document are clearly distinguished from each other using different text mark-ups.

This specification is divided into the following chapters:

At first, in Chapter 2 there are some definitions on the notation.

In Chapter 3 basic concepts about OSCI Transport are explained.

Chapter 4 presents the security mechanisms of OSCI Transport.

In Chapter 5 the reaction rules are defined and possible acknowledgements are listed.

Chapter 6 describes the structure of OSCI messages, contains the relevant schemas and defines value guidelines.

Finally in chapter 7 the conditions that software systems have to fulfill are listed in order to conform to this specification.

2. Notation Conventions

If not explicitly specified in a different way, this specification uses the following namespace binding as a basis:

Namespace Prefix	Namespace
ds	http://www.w3.org/2000/09/xmldsig#
osci	http://www.osci.de/2002/04/osci
soap	http://schemas.xmlsoap.org/soap/envelope
xenc	http://www.w3.org/2001/04/xmlenc#
xsd	http://www.w3.org/2001/XMLSchema

XML attributes appear unqualified if no namespace prefix has been indicated explicitly.

3. Messaging via OSCI Transport

In this chapter the basic concepts of OSCI Transport are described.

First, in Section 3.1 the entities are listed that are involved in communication by means of OSCI Transport.

Section 3.2 describes the levels information is exchanged on in OSCI Transport.

Section 3.3 explains the concept of the dialog, Section 3.4 explains the concept of the delivery.

Finally in Section 3.5 four basic scenarios for communication by means of OSCI Transport are introduced.

3.1 Entities Involved in Communication

OSCI Transport is a protocol with which basically any kind of information can be exchanged between users.

Here the term of a *user* is defined very widely. For example a user can be a person, a group of persons or even a software system. OSCI Transport does not have its own user administration, but works with an open user group. The only condition that an entity has to fulfill in order to be able to function as a user in OSCI Transport is to possess an encryption certificate. Encryption certificates are used for internal addressing in OSCI Transport.

In OSCI Transport two users never communicate directly, but always using a further entity called the *intermediary*. The intermediary functions as a communication agent, records the message flow, checks certificates and can perform other value-added services.

The intermediary's role is neutral. However, it can be physically located at one of the users.

In OSCI Transport specific users appear in the role of a *service provider*. Apart from an encryption certificate a service provider needs a URL, under which he is accessible online (in principle all the time).

3.2 Communication Levels

In OSCI Transport information is exchanged on three levels. Depending on the level, the entities involved assume different roles. In the following the individual levels, the roles that the entities involved assume and the exchanged data are described (cf. Figure 1).

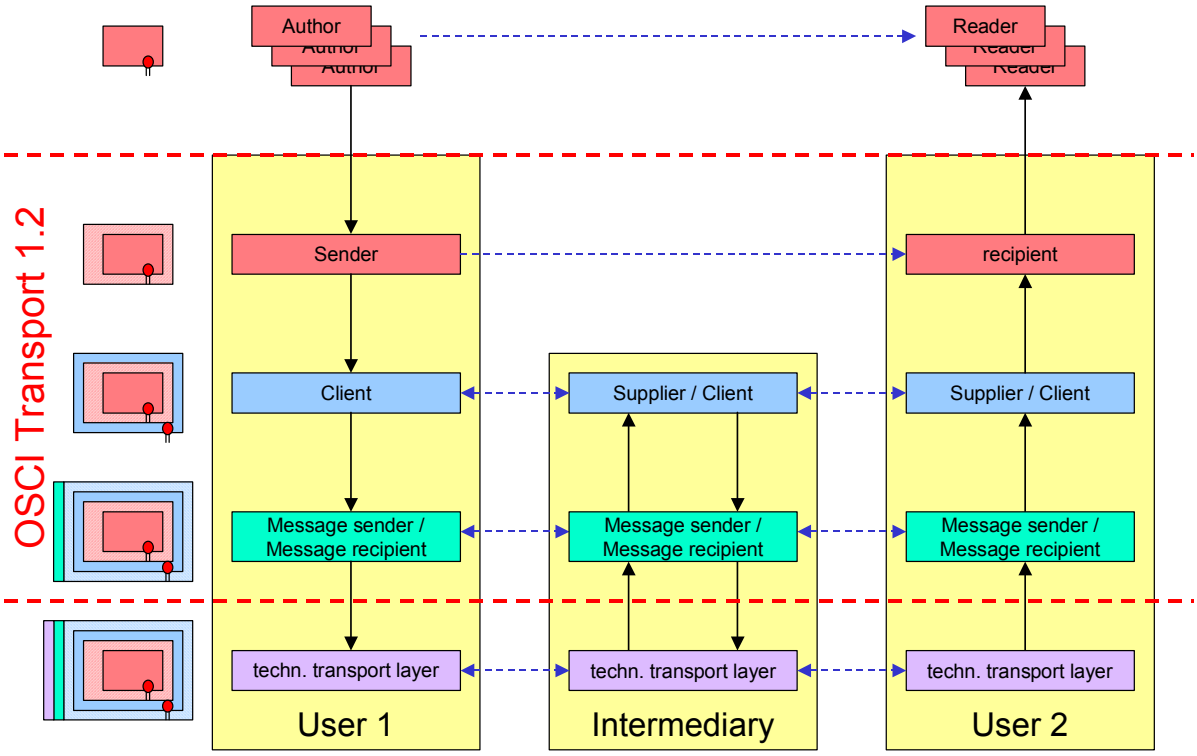

Figure 1: Communication levels

- *Business transaction level*: One or several *authors* put information (so-called *content data,* cf. [OSCI-ANF]) at the disposal of one or several *readers*. OSCI Transport does not make any requirements on the kind or the structure of the content data. In principle the content data can deal with any subject. Content data are transmitted in a *delivery* from a *sender* to a *recipient*. The roles of the sender and recipient are both assumed by users. The intermediary does not appear on the business transaction level.

  OSCI Transport does not say anything about the relationship that exists between authors and senders or readers and recipients. The data transport that might be necessary from the authors to the sender and from the recipient to the readers is not part of the application field of this specification.
- *Request level*: A *client* directs a *request* to a *supplier.* The request contains a working instruction and further information the supplier needs to execute the instruction and later send a *response* to the client. In requests and responses especially deliveries can be transmitted.

  Usually users act as clients and the intermediary as supplier. There are, however, also scenarios where the intermediary assumes the role of a client and a user (who then will be a service provider) assumes the role of a supplier.
- *Message level*: A *message sender* sends an *OSCI message* to a *message recipient*. Both user and intermediary can assume both roles. In that case one of the communication partners (message sender or message recipient) has to be an intermediary; users never communicate directly.

An OSCI message contains either exactly one request or exactly one response or exactly one error message. The request / the response can be encrypted. In this case the OSCI message contains additional information the message recipient needs to decrypt the request / response.

3.3 Dialogs

The term *dialog* deals with the request level of OSCI Transport. Request-response pairs are put together to dialogs. Each dialog contains at least one request-response pair.

A dialog exists exactly between one client and one supplier. It can come into being by two different ways:

- *explicitly* by a special dialog init request of the client to the supplier. The dialog will last at most until the supplier will have sent the response upon a specific dialog exit request. When the supplier announces an error that leads to a dialog abort the dialog will be cancelled prematurely.
- *implicitly* by a request that is different from the dialog init request. The dialog consists exactly of this request and its corresponding response.

Within an explicit dialog a client may send a subsequent request only when he has received the supplier's response to the previous request.

If an explicit dialog has been opened and the supplier does not receive a further request of the client after a certain period of time (which he can define freely), then he will close the dialog assuming that the client has not or not completely received the latest response.

A dialog is identified by the *ConversationId* assigned to it. The ConversationId is assigned by the supplier. It is definite as a supplier assigns each ConversationId only once. The ConversationIds assigned by different suppliers, however, do not necessarily have to be different in pairs.

Within one dialog, client and supplier number their messages independently from each other starting with 0. The number of the corresponding request / response is called *SequenceNumber*.

The terms *ConversationId* and *SequenceNumber* are borrowed from [EBXML].

Any request and any response can be identified uniquely by the data of the supplier, the ConversationId and the SequenceNumber.

3.4 Deliveries

A *delivery* contains all the data that are transmitted by a user acting as a sender to a user acting as a recipient. Thus, a delivery contains particularly content data.

For content data written in XML, OSCI Transport offers so-called *content data containers.* Other content data are packed into a*ttachments*.

Content data containers can be signed by the authors. The signing of attachments is also possible using an indirection: For that a hash value is calculated and put into a content data container, which is subsequently signed.

OSCI Transport offers an end-to-end encryption on the level of the content data. That means, content data containers and attachments can be encrypted so that they can be seen by the readers only.

Apart from the content data a content data container can contain encryption information for further content data containers and attachments that are part of the same delivery.

For example, to guarantee that two readers can get common access to the transmitted data (multiple encryption, "four-eye-principle“), it is possible to proceed in the following way:[1]

The actual data are put into a content data container 2. The latter is encrypted using the public key of reader 2. The pertaining encryption information (in particular the encrypted session key) is then packed into a content data container 1, which is encrypted using the public key of reader 1. Included in the delivery will be finally the encryption information for content data container 1 and the two encrypted content data containers. Figure 2 shows this in the form of a diagram.

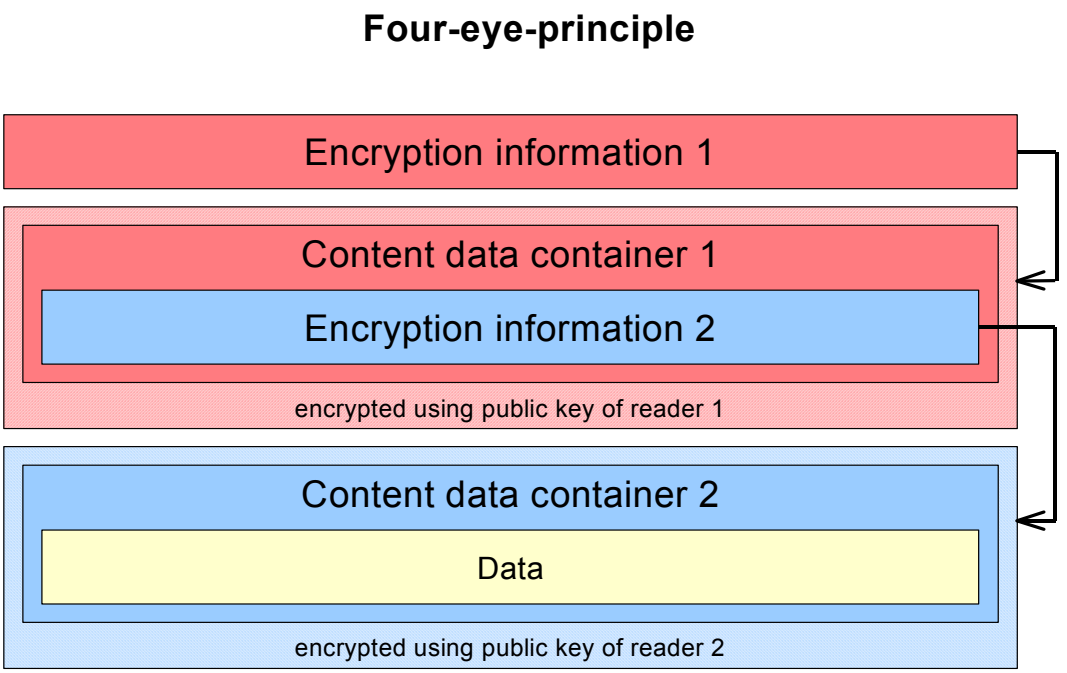

Figure 2: Possible realisation of the four-eye-principle

Reader 2 cannot see the data of content data container 2 without the help of reader 1. The session key used for encrypting content data container 2 has been encrypted using his public key, but the encrypted session key is in content data container 1, which can be decrypted by reader 1 only.

Reader 1, too, needs the help of reader 2 to be able to look at the data in content data container 2. The session key used for encrypting content data container 2 is in content data container 1, but the session key has been encrypted using the public key of reader 2, so that it can only be decrypted by reader 2.

[1] The realisation of a multiple encryption or "four-eye-principle" is the responsibility of the corresponding specific process. The method shown represents only an example of multiple encryption using the means of OSCI Transport.

Reader 1 and reader 2 have to work together to be able to see the data. First reader 1 has to decrypt content data container 1, so that reader 2 can get access to the encryption information for content data container 2. Reader 2 will then decrypt content data container 2.

In this way, also rule mechanisms can be realised in OSCI Transport. Here, along with the information they need in order to decrypt content data containers and/or attachments, the readers receive rules that restrict the access to these content data containers and/or attachments. For example in a content data container 1 could be stored the encryption information for two content data containers 2 and 3 as well as the following rule: "Do not open content data containers 2 and 3 before a certain point in time“. Since a reader is not able to access the encryption information for content data containers 2 and 3 until he will have opened content data container 1, he has to get to know the rules first to be able to decrypt content data containers 2 and 3. Figure 3 shows this in the form of a diagram.

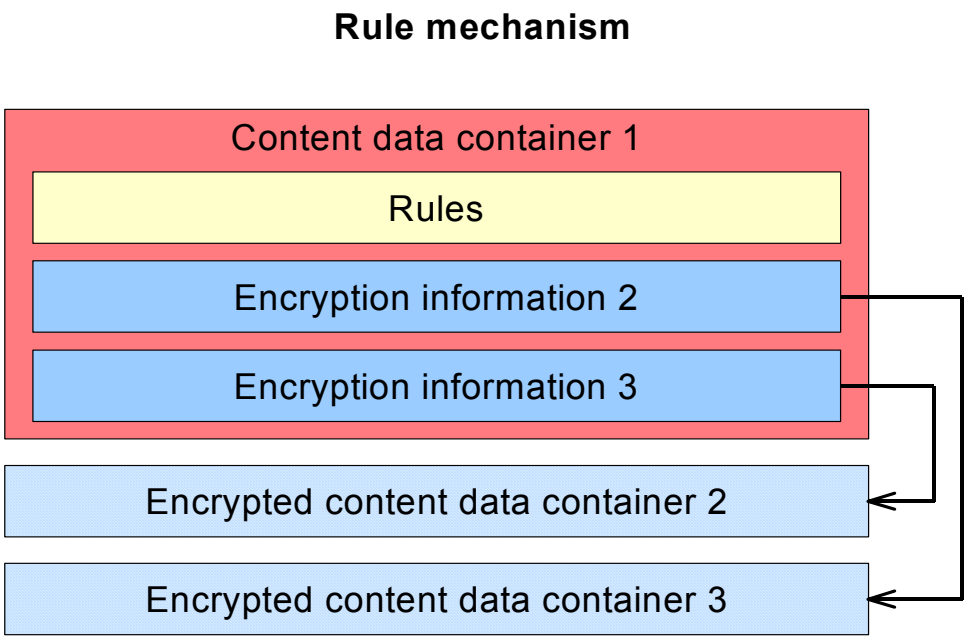

Figure 3: Possible realisation of a rule mechanism

A so-called *MessageId* can be assigned to a delivery. This term was borrowed from [EBXML].

The MessageId refers to exactly one delivery on the way from the sender via the intermediary to the recipient. It is uniquely definite in terms of place and time all over the world.

It will be generated by the intermediary upon request of the sender.

3.5 Scenarios

OSCI Transport supports four different scenarios for the exchange of content data between users. The individual scenarios differ from each other in three different aspects:

- Is merely a delivery transmitted from a sender to a recipient (one-way message) or does the recipient react directly upon the reception of a delivery sending another delivery to the sender (request-response)?
- Does the recipient have to actively collect a delivery at the intermediary (active recipient) or is it forwarded automatically to him without his being involved (passive recipient)?
- Does the content data exchange have to be recorded in a retraceable way or is that not required?

The communication scenario to be used is determined by the client.

In the following the scenarios are described. In Section 6.6 the request and response types mentioned are explained in detail.

3.5.1 One-Way Message, Active Recipient, Recording

In this scenario content data are transmitted from a user 1 to a user 2. User 2 has to be active to get the delivery. The transmission is recorded in a retraceable way.

This scenario presents itself, when it cannot be guaranteed that user 2 will permanently be available. In this scenario, user 2 does not necessarily need to have a URL. In other words, he does not have to be a service provider.

The rough process is as follows:

1. User 1 sends a MessageId request to the intermediary.
2. The intermediary generates a new MessageId and sends it within a MessageId response to user 1.
3. User 1 sends a delivery request to the intermediary. The delivery request contains a delivery which will be identified by the MessageId just received.
4. The intermediary generates a process card of the delivery just received and records the reception there. It sends a delivery response to user 1.
5. User 2 sends a dialog init request to the intermediary.
6. The intermediary reacts sending a dialog init response to user 2.
7. User 2 sends a fetch delivery request to the intermediary by which he asks for the deliveries that are there waiting for him.
8. The intermediary sends a fetch delivery response to user 2, which contains the delivery of user 1, and records the forwarding of the delivery on the process card.
9. User 2 sends a dialog exit request to the intermediary.
10. The intermediary records the reception of the delivery by user 2 on the process card and sends a dialog exit response to user 2.

Figure 4 contains a schematic presentation of the essential steps of this process.

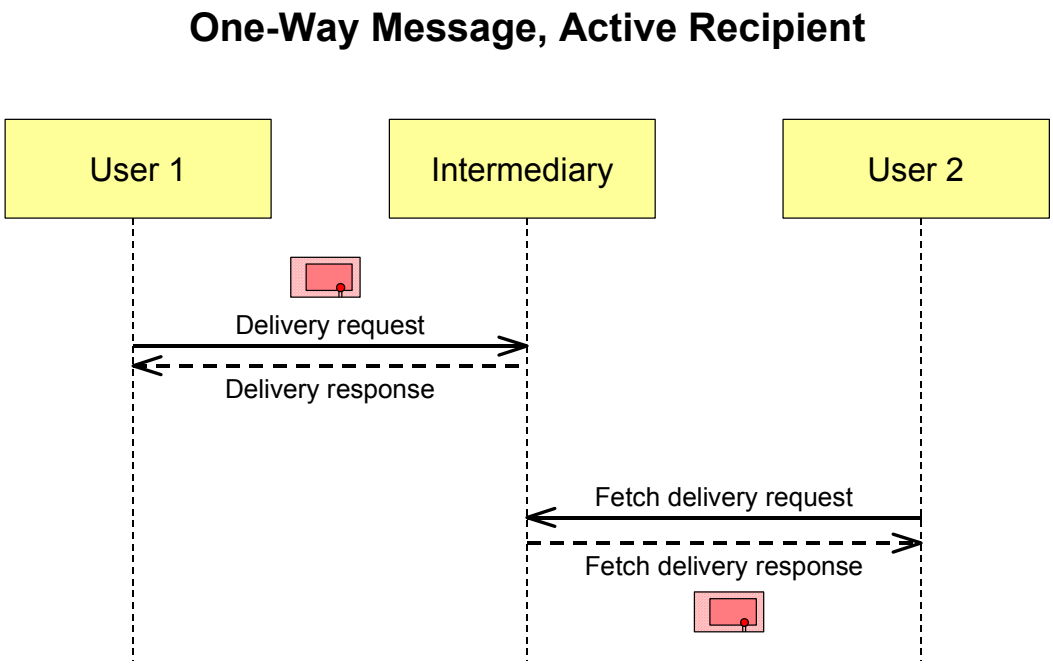

Figure 4: One-way message, active recipient

3.5.2 One-Way Message, Passive Recipient, Recording

In this scenario content data are transmitted from a user 1 to a user 2. The delivery is forwarded to user 2 without his being involved. The transmission is recorded in a retraceable way.

This scenario presents itself if user 2 is to receive the delivery without delay. In this scenario user 2 has permanently to be available under a URL. In other words, here user 2 is a service provider.

The rough process is as follows:

1. User 1 sends a MessageId request to the intermediary.
2. The intermediary generates a new MessageId and sends it within a MessageId response to user 1.
3. User 1 sends a forwarding delivery request to the intermediary. The forwarding delivery request contains a delivery, which is identified by the MessageId just received.
4. The intermediary generates a process card for the delivery just received and records the reception there. Along with the delivery it sends an acceptance request to user 2 and records the forwarding of the delivery on the process card.
5. User 2 sends an acceptance response to the intermediary confirming that he has received the delivery.
6. The intermediary records the reception of the delivery by user 2 on the process card and sends a forwarding delivery response to user 1.

Figure 5 contains a schematic presentation of the essential steps of this process.

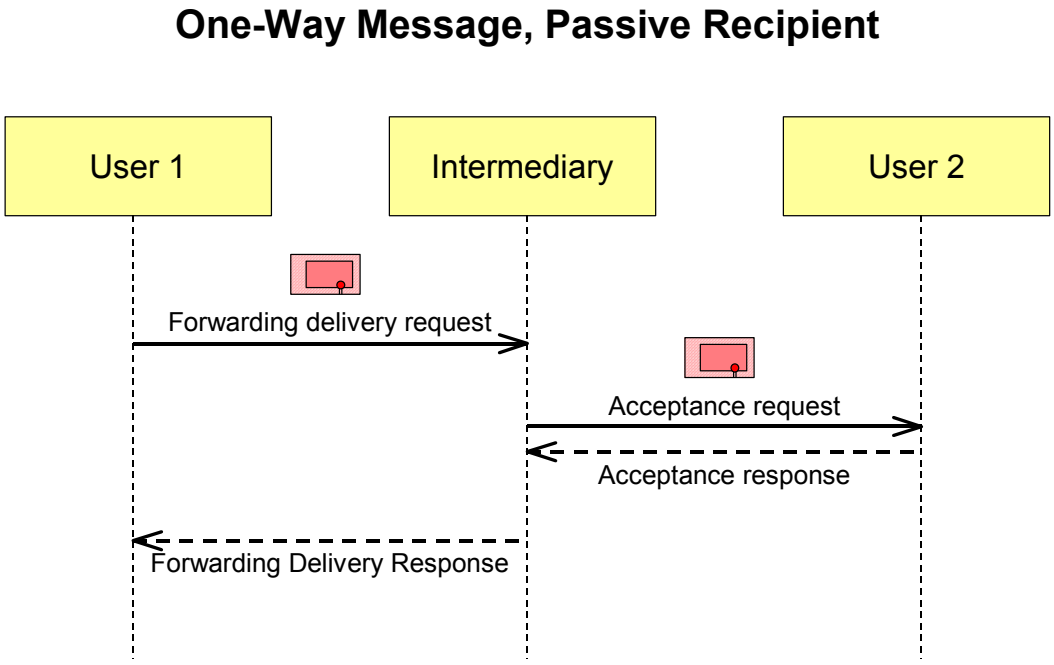

Figure 5: One-way message, passive recipient

3.5.3 Request-Response, Passive Recipient, Recording

In this scenario first a delivery 1 is transmitted from a user 1 to a user 2, then a delivery 2 is transmitted from user 2 to user 1. The transmission is recorded in a retraceable way.

This scenario presents itself if user 2 is to react immediately upon the delivery of user 1. In this scenario user 2 has permanently to be available under a URL. In other words, here user 2 is a service provider.

The rough process is as follows:

1. User 1 sends a MessageId request to the intermediary.
2. The intermediary generates a new MessageId and sends it within a MessageId response to the user 1.
3. User 1 sends a dialog init request to the intermediary.
4. The intermediary reacts sending a dialog init response to user 1.
5. Together with delivery 1 user 1 sends a mediate delivery request to the intermediary.
6. The intermediary generates a process card 1 for delivery 1, recording the reception there. Along with delivery 1 it sends a processing request to user 2, recording the forwarding of delivery 1 on process card 1.
7. User 2 sends a MessageId request to the intermediary.

8. The intermediary generates a new MessageId and sends it within a MessageId response to user 2.
9. Along with delivery 2 user 2 sends a processing response to the intermediary.
10. The intermediary generates a process card 2 for delivery 2, recording the reception there. Furthermore, it records the reception of delivery 1 by user 2 on process card 1. Along with delivery 2 it sends a mediate delivery response to user 1.
11. User 1 sends a dialog exit request to the intermediary.
12. The intermediary records the reception of delivery 2 by user 2 on process card 2 and sends a dialog exit response to user 1.

Figure 6 contains a schematic presentation of the essential steps of this process.

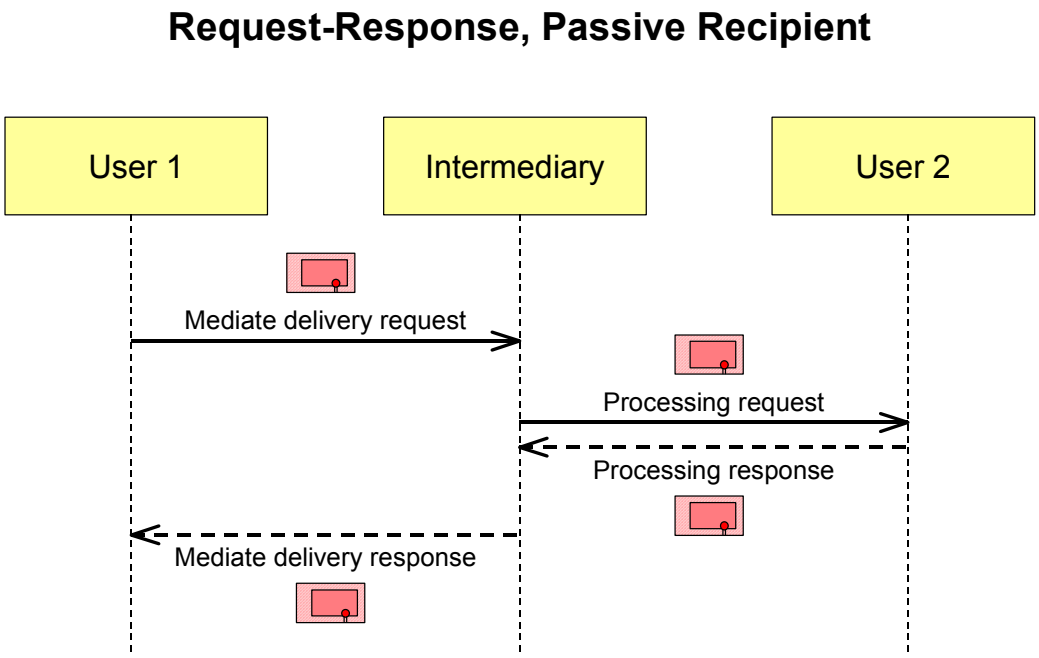

Figure 6: Request-response, passive recipient

3.5.4 Request-Response, Passive Recipient, No Recording

Just as in the previous scenario, here first a delivery 1 is transmitted from a user 1 to a user 2, then a delivery 2 is transmitted from user 2 to user 1. However, in this scenario the transmission is *not* recorded.

This simple scenario is intended for cases in which none of the parties has to prove after the communication has taken place that they have sent or received one of the deliveries and for cases in which a possible double submission of one of the deliveries does not cause any problem.

The rough process is as follows:

1. User 1 sends a dialog init request to the intermediary.

2. The intermediary reacts sending a dialog init response to user 1.
3. Along with delivery 1 user 1 sends a mediate delivery request to the intermediary.
4. Along with delivery 1 the intermediary sends a processing request to user 2.
5. Along with delivery 2 user 2 sends a processing response to the intermediary.
6. Along with delivery 2 the intermediary sends a mediate delivery response to user 1.
7. User 1 sends a dialog exit request to the intermediary.
8. The intermediary reacts sending a dialog exit response to user 1.

4. Security Mechanisms

OSCI Transport has at its disposal a number of mechanisms to secure data exchange. In this chapter the different mechanisms are described. Content and headlines of the individual sections are oriented towards [OSCI-ANF, chapter 6]. However, instead of the more general term of *usage data* used there, here we speak of *requests* and *responses*, as these are often to be dealt with differently.

4.1 Digital Signatures

Both content data and request data can be signed digitally. For that the data are processed according to [XDSIG].

A signature is generated by calculating a hash value for each XML element that has to be secured by the signature. The hash values are collected in a special XML element. For this element another hash value is calculated which first is specially formatted and then is encrypted with the private key of the person who signs.

A signature is checked by calculating and formatting again the hash value for the XML element the hash values for the protected elements have been collected in. The result has to coincide with the decrypted signature. In addition, all the hash values for the protected elements are recalculated and compared with the signed hash values.

In the following table the algorithms are listed that may be used for the generation of hash values in OSCI Transport.

Hash Algorithm	**Algorithm Identifier**
SHA-1	`http://www.w3.org/2000/09/xmldsig#sha1` [XDSIG, Section 6.2.1]
RIPEMD-160	`http://www.osci.de/2002/04/osci#ripemd160`

The signature algorithms supported by OSCI Transport are listed in the following table.

Signature Algorithm	**Algorithm Identifier**
Signature scheme RSASSA-PKCS1-v1_5 [PKCS 1] with hash algorithm SHA-1. Modulus length of the RSA key pair: min. 1024 bit	`http://www.w3.org/2000/09/xmldsig#rsa-sha1` [XDSIG, Section 6.4.2]
Signature scheme RSASSA-PKCS1-v1_5 [PKCS 1] with hash algorithm RIPEMD-160. Modulus length of the RSA key pair: min. 1024 bit	`http://www.osci.de/2002/04/osci#rsa-ripemd160`

4.1.1 Signing Content Data

Signatures on the business transaction level are generated by the authors.

4.1.2 Verifying Content Data Signatures

Readers can check signatures on the business transaction level for their own good but are not obliged to do so. The public key used to verify the signature comes from a certificate, which is transmitted together with the signed data. The reaction to a missing or invalid signature is at the readers' discretion.

4.1.3 Signing Requests and Responses

Requests and responses can (independently on or in addition to a possibly existing signature on the content data level) be signed in order to ensure their authenticity and integrity.

Signatures for requests are generated by the client.

Signatures for responses are generated by the supplier.

4.1.4 Verifying Request Signatures

If a request is signed, the supplier is obliged to perform a signature verification. The public key used for the signature verification comes from a certificate that is transmitted within the request. The supplier has the right to reject unsigned requests. Requests whose signatures are invalid are to be rejected. The rejection is announced to the client immediately in the response. Requests for which the supplier realises that they have been signed using a private key that – according to the certificate – must not be used for signing have to be rejected, too.

4.1.5 Verifying Response Signatures

The signature of a response can be verified by the client.

The public key used for the signature verification comes from a certificate that is transmitted within the response.

Two cases are to be distinguished:

- The client is an intermediary. This is the case, if within the execution of a forwarding or mediate delivery request of a user it has directed an acceptance request or processing request to a service provider.
- The client is a user. This is true in all the other cases.

An intermediary acting as a client has to check the signatures of responses. If it realises that the signature of a response is invalid or has been generated using a private key for which the certificate determines that it must not be used for signing, then it has to consider the response to be manipulated. It then will give an according acknowledgement to the user who sent it the forwarding delivery request or mediate delivery request.

If the signature is missing, the user will receive a corresponding warning.

A user acting as a client can check the signatures of responses for his own good, but is not obliged to do so. If he realises that the signature of a response is invalid or has been generated using a private key that – according to the certificate – must not be used for signing, and if an explicit dialog is opened, then the user must not send any further request (not even a dialog exit request) in this dialog.

The user will assume that he has not received the response the supplier has sent him. If the supplier does not receive a further request of the user within a certain period of time, he will

close the dialog assuming that the user has not received or has not received completely the latest response (cf. Section 3.3).

The reaction to a missing signature for responses is at the user's discretion.

4.2 Encryption and Decryption

Both content data and requests/responses can be encrypted. Responses to dialog init requests are obligatorily to be encrypted (cf. Section 4.6).

The encryption is performed according to [XENC] applying a hybrid encryption algorithm.

First a random session key is generated for a symmetric encryption algorithm. Using this key the actual data are encrypted. In a second step the session key is encrypted with the public encryption key of the addressee[2]. The encrypted data and the encrypted session key are forwarded to the addressee.

The addressee first decrypts the session key using his private encryption key. Using the decrypted session key he is able to decrypt the actual data.

The symmetric encryption algorithms that can be used in OSCI Transport to encrypt the actual data are put together in the following table.

Encryption Algorithm	**Algorithm Identifier**
Two-Key-Triple-DES	`http://www.w3.org/2001/04/xmlenc#tripledes-cbc` [XENC, Section 5.2.1]
AES-128	`http://www.w3.org/2001/04/xmlenc#aes128-cbc` [XENC, Section 5.2.2]
AES-192	`http://www.w3.org/2001/04/xmlenc#aes192-cbc` [XENC, Section 5.2.2]
AES-256	`http://www.w3.org/2001/04/xmlenc#aes256-cbc` [XENC, Section 5.2.2]

[XENC] contains information, how these algorithms have to be used: Especially, they have to be applied in CBC mode, an initializing vector filled with random numbers is to be used, and padding is to be performed in a specific way (padding according to [PKCS 5] represents a special case of the process described in [XENC]).

It has to be ensured that not the same session key is used for data that are directed to different addressees.

The encryption of the session key is performed by means of RSAES-PKCS1-v1_5 [PKCS 1] (algorithm identifier: `http://www.w3.org/2001/04/xmlenc#rsa-1_5` [XENC, Section 5.4.1]). The modulus length of the RSA key pair used has to be at least 1024 bit.

[2]The *addressee* in case of requests is the supplier, in case of responses the client, and in case of content data a reader.

4.2.1 Encryption of Content Data

Content data are encrypted by the authors. The readers' public keys used for encryption come from certificates. The way how the authors obtain access to these certificates is not described in this specification.

4.2.2 Decryption of Content Data

With the help of certificates, which are enclosed with the encrypted content data, the readers can find out, which of their private keys are needed for the decryption.

The reaction to content data that cannot be decrypted is at the readers' discretion.

4.2.3 Encryption of Requests

Requests are encrypted by the client. The public key of the supplier used for encryption comes from a certificate. The way how the client obtains access to this certificate is not described in this specification.

4.2.4 Decryption of Requests

With the help of a certificate, which is enclosed with the encrypted data, the supplier can find out, which of his private keys is needed for the decryption.

If the certificate determines that the public key included must not be used for encryption, the supplier is not obliged to decrypt the request. In this case the supplier does not have to send a response.

If a supplier is not able to decrypt a request, he will not be obliged to answer the request either.

The reason for this is that in this case the client who sent the request cannot be detected by means of OSCI Transport.

4.2.5 Encryption of Responses

If necessary, responses are encrypted by the supplier.The public key of the client used for encryption comes from a certificate. The supplier takes this certificate

- from the dialog init request, if the request was sent within an explicit dialog, or
- from the request the response to be encrypted refers to.

4.2.6 Decryption of Responses

With the help of a certificate, which is enclosed with the encrypted data, the client can find out, which of his private keys is needed for decryption.

If a client cannot decrypt a request and if an explicit dialog has been opened, the client must not send any further request (not even a dialog exit request) in this dialog.

Otherwise the supplier would assume that the client received the response and was able to interpret it.

4.3 Recording Times

The intermediary holds a so-called *process card* for each delivery. On the process card it records especially the following times:

1. Time of presentation at the intermediary

   From the technical point of view, this is the time when the intermediary realises that it has received a delivery request, forwarding delivery request or mediate delivery request.

2. Time of forwarding to the recipient

   From the technical point of view this is the time when the intermediary has completely assembled the message that contains the delivery to the recipient. Especially this time can differ from the time of the technical process of forwarding.

3. Time of acknowledging the receipt by the recipient

   From the technical point of view this is exactly one of two times:

   1. the delivery has been sent together with an acceptance request or processing request: the time when the intermediary realises that the recipient has sent a positive acknowledgement;
   2. the delivery has been sent together with a fetch delivery response: the time when the intermediary realises that it has received another request within the same explicit dialog the delivery has been transmitted in.

Upon request of the sender the intermediary can generate cryptographic time stamps for these times.

In case 1 the time stamp refers to the delivery request, forwarding delivery request or mediate delivery request of the sender.

In case 2 the time stamp refers to the processing request or acceptance request, or by the fetch delivery response of the intermediary.

In case 3.1 the time stamp refers to the response received from the recipient.

In case 3.2 the time stamp refers to the subsequent request received from the recipient.

Concerning the format of cryptographic time stamps this specification does not make any prescriptions.

Note, however, that particularly time stamps according to [ISIS-MTT, Part 4] can be used.

4.4 Certificate Checks

As certificate format OSCI Transport supports X.509v3 in the form described in [ISIS-MTT, Part 1].

The intermediary has to check all the certificates that are located on the request level in a message received. Furthermore the intermediary has to check all the certificates of the pertaining certificate chains.

The following checks have to be performed:

- Mathematical check of the certificate signature: The hash value over the certificate is recalculated and formatted. The result has to coincide with the decrypted signature of the certificate.
- Off-line validity check: It is checked that the time of the check is within the validity period mentioned in the certificate.
- Online validity check: It is checked that the certificate has not been revoked at the time of the check.

The intermediary records the result of each of these checks in a check protocol. If the intermediary is not able to perform one of these checks (for example because it does not know the certificate authority), it will record that into the check protocol, too.

Within one dialog each certificate needs to be checked only once.

The intermediary has to reject a delivery request, forwarding delivery request or mediate delivery request, if one of the readers' encryption certificates or the encryption certificate of the user who acts as a supplier on the request level has been revoked. The client receives a corresponding acknowledgement immediately in the response upon the corresponding request.

If the intermediary realises that the encryption certificate of the client has been revoked, it will not be allowed to encrypt his response using this certificate. In this case the client will receive a not encrypted response that only points to this fact.

In other words, especially the dialog init requests of a client are rejected, about whose encryption certificate the supplier knows that it has been revoked.

Before the process of encryption, senders and authors should perform at least the checks that are possible to perform off-line, for the encryption certificate of the corresponding recipient / readers. If one of these checks has a negative result, this certificate must not be used for encryption.

After having received signed data, recipient and readers should perform at least the checks that are possible to perform off-line, for the signature certificates of the corresponding senders / authors. The reaction to a negative result of one of these checks is at the recipient's / readers' discretion.

4.5 Challenge-Response Process

OSCI Transport uses a challenge-response process in order to guarantee that the entities involved in a dialog (client and supplier) will not change during a dialog.

Along with each request the client sends a freely chosen value. The supplier has to repeat this value as a response value in his response.

If a client receives an invalid response value and if an explicit dialog has been opened, the client must not send any further request (not even a dialog exit request) in this dialog.

The client will assume that he has not received the response the supplier has sent him. If the supplier does not receive another request from the client in a certain period of time (that he can freely determine), he will close the dialog, assuming that the client has not or not completely received the latest response (cf. Section 3.3).

Outside an explicit dialog the reaction to an invalid response value is at the client's discretion.

Within an explicit dialog the supplier sends along with each response (starting with the response to a dialog init request and ending with the response to the request that precedes the dialog exit request) a freely chosen value as a challenge value. The client has to repeat this value as a response value in his following request.

If the supplier receives an invalid response value, he has to reject the corresponding request.

To maintain this process functioning the challenge value has to be changed in each message. It must not be derivable simply from information accessible to unauthorised users (e.g. time).

Apart from the response value of the client, the supplier checks the ConversationId and the SequenceNumber. If the supplier receives a request with an unexpected ConversationId or SequenceNumber, he has to reject the request.

4.6 User Authentication within the Explicit Dialog Initialization

Within the explicit dialog initialization, the supplier checks that the client possesses the private key that belongs to the encryption certificate the client has sent him along with the dialog init request.

To this end the supplier encrypts his response using the public key of this certificate.

If the client does not possess the private key belonging to it, he will not be able to decrypt the dialog init response. Consequently he cannot obtain access especially to the challenge value in the dialog init response. Since the intermediary (as is described in Section 4.5) has to reject any subsequent request within the dialog that has no valid triple consisting of ConversationId, SequenceNumber and Response, the client will not be able to hand in subsequent requests within this dialog.

Furthermore, within an explicit dialog, in order to encrypt his responses to the client, the supplier will always use the public key from the encryption certificate communicated to him by the client in the dialog init request.

This way it can be prevented that a client will change his encryption certificate in the course of the dialog.

4.7 Checking MessageIds

The MessageId check helps to avoid double submission of deliveries.

In delivery requests and forwarding delivery requests a client has to state a MessageId along with the delivery. In mediate delivery requests the stating of a MessageId is optional.

If in the mentioned requests a MessageId is stated, the intermediary has to check, that

- it has generated the MessageId
- the MessageId has not already been used

If one of these checks has a negative result, the intermediary has to reject the request the sender has handed in the delivery with.

The intermediary may reject the request also for practical reasons, if the generation of the MessageId took place already some time ago (in the range of a time-out during an explicit dialog).

If a sender does not state any MessageId in a mediate delivery request to the intermediary, the recipient will be allowed to reject the resulting processing request of the intermediary with the note that it does not accept any delivery without MessageId. The sender will receive a corresponding acknowledgement within the mediate delivery response of the intermediary.

4.8 Receipt Mechanisms

OSCI Transport knows of two explicit and two implicit receipt mechanisms.

The explicit receipt mechanisms are:

- At any time sender and recipient of a delivery can ask the intermediary for copies of the corresponding process card.
- In each response to a fetch request the selection criteria that the client has stated in his request are repeated.

The implicit receipt mechanisms are:

- By sending a response with an acknowledgement code that indicates success a supplier confirms that he received the corresponding request and was able to execute it.
- By sending a request with a response value that coincides with the challenge value of the preceding response and whose SequenceNumber is by 1 higher than the SequenceNumber of the preceding request, a client confirms that he received the response to the preceding request.

5. Reaction Rules and Acknowledgements

When a supplier receives a message, he has to take the following steps:

1. Perform XML schema and other syntactic checks on the message level.
2. If applicable, decrypt the request data.
3. Perform XML schema and other syntactic checks on the request level.
4. Determine and check ConversationId, SequenceNumber and Response.
5. If the response is to be encrypted and if the encryption certificate of the client has not been checked yet in this dialog, determine and check it.
6. If applicable, check the signature.
7. Check the other certificates contained in the request that have not been checked yet in this dialog.
8. Execute the request.

The supplier is obliged to perform steps 5 to 7 only if he acts as an intermediary.

Each one of these steps can be performed successfully, can end with a warning or can fail. If one of these steps fails, the supplier has to cancel the process at the corresponding point. If there are warnings, the supplier can continue.

In his response to the client the supplier transmits acknowledgements that demonstrate which of the steps he could execute, what type of warnings have occurred and which of the steps have failed.

In order to enable an automatic interpretation of the acknowledgements by the client, each acknowledgement consists of a four-figure numeric code which is formed according to the following pattern:

- The first digit of the code indicates the kind of acknowledgement. `0` stands for a success message, `3` stands for a warning and `9` stands for an error message.
- The second digit indicates the step the acknowledgement is given for (if the step is unknown, this digit is `0`).
- The last two digits of the code form an indenture number.

The acknowledgements are put into the response in the sequence of their generation.

So on the basis of the last acknowledgement in the response the client is able to recognise if the supplier could execute the request correctly or if errors occurred:

- If the last acknowledgement is a success message or a warning, this means that the request was executed. Then in all the steps there could have been warnings at most.
- In contrast, if the last acknowledgement is an error message, this means that an error occurred.

In normal case the supplier communicates acknowledgements to the client on the request level (cf. Section 5.2). Only if it is not possible to communicate acknowledgements to the client on the request level, they can be sent on the message level (cf. Section 5.1).

If one of the steps 1 to 5 fails, the supplier will not be obliged to send a response.

The reason for this is that in these cases it is impossible to identify the client who sent the request by means of OSCI Transport.

If, however, the technical protocol that is used as a basis allows to send an acknowledgement (this is the case for example for HTTP), the supplier should do that.Then the client will receive an error message on the message level.

To guarantee the interoperability, modifications and additions to the acknowledgement codes may only be made in agreement with the editor.

5.1 Error Messages on the Message Level

Acknowledgements are communicated on the message level only if either the client who has sent the request could not have been detected or if an internal error has occured at the supplier's that makes it impossible to sign or encrypt the response or insert it into an OSCI message. So acknowledgements on the message level are always error messages.

Acknowledgements on the message level are transmitted in a `soap:Fault` element in the SOAP body (cf. [SOAP 1, Section 4.4]). The `soap:Fault` element consists of exactly three elements: `faultcode`, `faultstring` and `detail`.

The possible values of `faultcode` are described in [SOAP 1, Section 4.4.1].

Of these values OSCI Transport only uses `soap:Client` and `soap:Server`.

In the element `faultstring` a plain text description of the error is given. The exact formulation depends on the national language used and is up to the supplier.

The element `detail` contains an element `osci:Code` that includes the numeric code determining the occurred error.

The values of `faultcode` and `osci:Code` in the different error situations can be seen in the following table.

Step	Error situation	faultcode	osci:Code
1	Received message is no valid OSCI message	`soap:Client`	`9100`
2	Supplier does not have the private key for the encryption certificate on the message level at his disposal	`soap:Client`	`9200`
2	Encryption certificate on the message level has been revoked	`soap:Client`	`9201`
2	Request data could not be decrypted	`soap:Client`	`9202`
3	Request data do not represent a valid request	`soap:Client`	`9300`
4	ConversationId or SequenceNumber or response value are not correct	`soap:Client`	`9400`
5	Encryption certificate of the client is not included in the request data	`soap:Client`	`9500`
5	Signature on the encryption certificate of the client is invalid	`soap:Client`	`9501`
5	Encryption certificate of the client has been revoked	`soap:Client`	`9502`

Step	Error situation	faultcode	osci:Code
5	Internal error at the supplier's when determining or checking the encryption certificate	soap:Server	9503
unknown	Internal error at the supplier's	soap:Server	9000

Only if the element faultcode has the value soap:Server the client is allowed to try to resend the unchanged message.

5.2 Acknowledgements on the Request Level

Usually the communication of the supplier's acknowledgements to the client takes place on the request level.

The acknowledgements are transmitted in a special element osci:Feedback whose inner elements have the following values:

- The element osci:Code contains the numeric code that denotes the acknowledgement.
- The element osci:Text contains a plain text concerning the acknowledgement. The exact formulation depends on the national language used and is up to the supplier.

The possible values of osci:Code are compiled in the following table:

Step	Situation	osci:Code
5	Encryption certificate of the client is invalid with regard to time	3500
5	Check of the client's encryption certificate could not be performed completely	3501
6	Request has not been signed, although this supplier demands signature for this request type	9600
6	Signature for request is invalid	9601
6	Not all of the required elements of the request have been signed.	9602
6	Internal error at the supplier's when checking the signature	9603
7	Signature certificate of the client is invalid with regard to time	3700
7	Signature over the signature certificate of the client is invalid	9700
7	Signature certificate of the client has been revoked	9701
7	Signature certificate of one of the authors is invalid with regard to time	3701
7	Signature over the signature certificate of one of the authors is invalid	3702
7	Signature certificate of one of the authors has been revoked	3703
7	Encryption certificate of one of the authors is invalid with regard to time	3704
7	Signature over the encryption certificate of one of the authors is invalid	9704
7	Encryption certificate of one of the authors has been revoked	9705
7	Encryption certificate of the recipient is invalid with regard to time	3705

Step	Situation	osci:Code
7	Signature over the encryption certificate of the recipient is invalid	9706
7	Encryption certificate of the recipient has been revoked	9707
7	Encryption certificate of one of the readers is invalid with regard to time	3706
7	Signature over the encryption certificate of one of the readers is invalid	9708
7	Encryption certificate of one of the readers has been revoked	9709
7	Certificate check could not be performed completely	3707
7	Internal error at the supplier's when performing the certificate check	9710
8	Acceptance of delivery without MessageId rejected	9800
8	Illegal MessageId (MessageId was not generated by this supplier or MessageId has already been used)	9801
8	There are further deliveries present for this client	3800
8	There are further process cards present that meet the criteria given in the request	3801
8	Request has not been sent within an explicit dialog, although this is demanded for this request type	9802
8	There is no delivery present that meets the criteria given in the request	9803
8	There is no process card present that meets the criteria given in the request	9804
8	Delivery of the sender has not been accepted by the recipient	9805
8	Delivery of the sender has been accepted by the recipient, but his response does not contain any delivery to the sender	9806
8	Signature of the recipient over the acceptance response or processing response is invalid	9807
8	Signature of the recipient over the acceptance response or processing response is missing	3802
8	Internal error at the recipient's	9808
8	Internal error at the intermediary's during forwarding or mediate delivery request. Delivery has not reached the recipient.	9809
8	Internal error at the intermediary's during forwarding or mediate delivery request. Delivery has reached the recipient.	9810
8	Internal error at the supplier's when executing the request	9811
8	Request executed, dialog closed	0800
8	Request executed, dialog continues to be open	0801

6. Message Structure

In this section the message structure in OSCI Transport is described.

First in Section 6.1 the namespace defined by this specification is stated.

Then Section 6.2 describes the meanings of the individual elements defined in this namespace.

Section 6.3 contains an XML schema which is detailed in the sections that follow.

The adaptions of XML Signature and XML Encryption for OSCI Transport can be found in Sections 6.4 and 6.5.

The structure of the individual request types and response types that exist in OSCI Transport is described in Section 6.6.

Finally in Section 6.7 the structure of OSCI messages is explained.

6.1 Versioning

The following namespace is assigned to OSCI Transport in its version 1.2:

`http://www.osci.de/2002/04/osci`

On the basis of this namespace a supplier, sender or reader can determine the version of OSCI Transport that a client, recipient or author is using.

The namespace mentioned above is linked to the namespaces of the standards OSCI Transport is based on:

- `http://schemas.xmlsoap.org/soap/envelope/` (SOAP)
- `http://www.w3.org/2000/09/xmldsig#` (XML Signature)
- `http://www.w3.org/2001/04/xmlenc#` (XML Encryption)

If for one of these standards an update version appears with a modified namespace and if OSCI Transport is to support this version, a new namespace will be published.

It is an explicit aim to support the consolidated versions of SOAP 1.2 and XML Encryption.

Furthermore the namespace is changed when acknowledgement codes change.

6.2 Meaning of Elements with Namespace http://www.osci.de/2002/04/osci

The following table states the meaning of each element with namespace `http://www.osci.de/2002/04/osci`.

For the sake of clarity the names of the elements are listed without their namespace prefixes.

Name	Meaning
`acceptDelivery`	Indicates an acceptance request
`CertType`	The attribute `Type` indicates the type of certificate: `unknown` / `advanced` / `qualified`
`Challenge`	Challenge value (cf. Section 4.5)

Name	Meaning
`CipherCertificateAddressee`	If along with the request or response a delivery is sent: encryption certificate of the recipient. Otherwise: encryption certificate of the user as the addressee of the request or response
`CipherCertificateIntermediary`	Encryption certificate of the intermediary. An intermediary has to state this element especially in case of each request it expects to receive an encrypted response upon and that is sent outside an explicit dialog. Within an explicit dialog to encrypt his responses the supplier always uses the encryption certificate that the intermediary has stated in the element `CipherCertificateIntermediary` of its dialog init request.
`CipherCertificateOriginator`	If along with the request or response a delivery is sent: encryption certificate of the sender. Otherwise: encryption certificate of the user as the sender of the request or response. A user has to state this element especially in case of each request that is sent outside an explicit dialog and that he expects to receive an encrypted response upon. Within an explicit dialog to encrypt its responses the intermediary always uses the encryption certificate that the user has stated in the element `CipherCertificateOriginator` of his dialog init request.
`CipherCertificateOtherAuthor`	Encryption certificate of an author who is not the sender of the delivery
`CipherCertificateOtherReader`	Encryption certificate of a reader who is not the recipient of the delivery
`ClientSignature`	Signature of a client for the request
`Code`	Acknowledgement code (cf. Chapter 5)
`ControlBlock`	Contains information with the help of which client and supplier are able to guarantee dialog integrity
`Content`	Contains the actual content data. Can especially contain links to attachments.
`ContentContainer`	Content data container
`ContentPackage`	Contains any number of content data containers. Some of them can be encrypted and are stored in elements `xenc:EncryptedData`.
`ContentReceiver`	Attribute URI indicates the URL of the service provider
`ConversationId`	ConversationId (cf. Section 3.3)
`Creation`	Time stamp *Time of presentation at the intermediary's*

Name	Meaning
CRL	Contains the date of the CRL that has been used for the online validity check
DesiredLanguages	Languages that the client prefers for textual acknowledgements. If possible, a supplier should take this information into consideration
exitDialog	Indicates a dialog exit request
Feedback	Acknowledgement of a supplier to a client (cf. Section 5.2)
fetchDelivery	Indicates a delivery request
fetchProcessCard	Indicates a fetch process card request
forwardDelivery	Indicates a forwarding delivery request
Forwarding	Time stamp *Forwarding to recipient*
getMessageId	Indicates a messageId request
initDialog	Indicates a dialog init request
Inspection	Entry in the check protocol of a certificate
InspectionReport	Check protocol for the certificates contained in a request
IntermediaryCertificates	Certificates of the intermediary
LDAP	Indicates that the online validity check was performed using LDAP
MathResult	The attribute Result indicates the result of the signature verification of the certificate
mediateDelivery	Indicates a mediate delivery request
MessageId	MessageId (cf. Section 3.4)
NonIntermediaryCertificates	Certificates of users, authors and readers
OCSP	Contains the OCSP response within the online validity check for a certificate
OfflineResult	The attribute Result indicates the result of the off-line validity check for the certificate
OnlineResult	The attribute Result indicates the result of the online validity check for the certificate
ProcessCard	Process card in which the intermediary records the way of a delivery from the sender to the recipient
ProcessCardBundle	Process card container
processDelivery	Indicates a processing request

Name	Meaning
QualityOfTimestamp	Indicates the desired quality of a time stamp on the process card. Depending on the value of the attribute actor, it refers either to the time stamp *Time of presentation at the intermediary's* (actor="http://www.osci.de/2002/04/osci#creation") or to the time stamp *Time of acknowledging the receipt by the recipient* (actor="http://www.osci.de/2002/04/osci#reception"). The desired quality is indicated in the attribute Quality. Possible values are: http://www.osci.de/2002/04/osci#cryptographic (a cryptographic time stamp is required) or http://www.osci.de/2002/04/osci#plain (system time is sufficient)
Quantity	The attribute Limit of this element indicates the maximum number of process cards to be returned
RecentModification	Time of the latest modification of a process card
Reception	Time stamp *Time of acknowledging the receipt by the recipient*
ReceptionOfDelivery	Time of presentation at the intermediary
ReplyProcessCardBundle	Process card container for the delivery in the mediate delivery response
RequestProcessCardBundle	Process card container for the delivery in the mediate delivery request
Response	Response value (cf. Section 4.5)
responseToAcceptDelivery	Indicates an acceptance response
responseToExitDialog	Indicates a dialog exit response
responseToFetchDelivery	Indicates a fetch delivery response
responseToFetchProcessCard	Indicates a fetch process card response
responseToForwardDelivery	Indicates a forwarding delivery response
responseToGetMessageId	Indicates a MessageId response
responseToInitDialog	Indicates a dialog init response
responseToMediateDelivery	Indicates a mediate delivery response

Name	Meaning
responseToProcessDelivery	Indicates a processing response
responseToStoreDelivery	Indicates a delivery response
SelectionRule	Selection criterion for deliveries or process cards
SequenceNumber	SequenceNumber (cf. Section 3.3)
SignatureCertificateAddresse	If along with the request or response a delivery is sent: Signature certificate of the recipient. Otherwise: Signature certificate of the user as the addressee of the request or response
SignatureCertificateIntermediary	Signature certificate of the intermediary. This element is present in each signed request and in each signed response of the intermediary
SignatureCertificateOriginator	If along with the request or response a delivery is sent: Signature certificate of the sender. Otherwise: Signature certificate of the user as the sender of the request or response. This element is present in all the requests and responses that contain deliveries with signed content data. Furthermore this element is in each signed request and in each signed response of a user.
SignatureCertificateOtherAuthor	Signature certificate of an author who is not the sender of a delivery
SignatureCertificateOtherReader	Signature certificate of a reader who is not the recipient of a delivery
storeDelivery	Indicates a delivery request
Subject	Subject of a delivery
SupplierSignature	Signature of a supplier for the response
Text	Acknowledgement text (cf. Section 5.2)
Timestamp	Time of the certificate check
X509IssuerName	Issuer of the checked certificate
X509SerialNumber	Serial number of the checked certificate

6.3 Global Type Definitions

For the global type definitions and abstract type templates the following schema is valid:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
```

```
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.w3.org/XML/1998/namespace"
   schemaLocation="http://www.w3.org/2001/xml.xsd" />
  <xsd:import namespace="http://www.w3.org/2000/09/xmldsig#"
   schemaLocation="http://www.w3.org/TR/2001/CR-xmldsig-core-
20010419/xmldsig-core-schema.xsd" />
  <xsd:import namespace="http://www.w3.org/2001/04/xmlenc#"
   schemaLocation="http://www.w3.org/TR/xmlenc-core/xenc-schema.xsd" />
  <xsd:import namespace="http://schemas.xmlsoap.org/soap/envelope/"
   schemaLocation="http://schemas.xmlsoap.org/soap/envelope/" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - general types and structures
      $RCSfile: order.xsd,v $, $Revision: 1.20 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### global simple types ### -->

  <xsd:simpleType name="LanguagesListType">
    <xsd:list itemType="xsd:language" />
  </xsd:simpleType>

  <xsd:simpleType name="MessageIdType">
    <xsd:restriction base="xsd:base64Binary">
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:simpleType name="Number">
    <xsd:restriction base="xsd:NMTOKEN">
      <xsd:pattern value="\d+" />
    </xsd:restriction>
  </xsd:simpleType>

  <!-- ### global complex types and templates ### -->

    <!-- ### common types ### -->

  <xsd:complexType name="CertificateType">
    <xsd:complexContent>
      <xsd:restriction base="ds:KeyInfoType">
        <xsd:sequence>
          <xsd:choice>
            <xsd:element ref="ds:X509Data" />
```

```
        </xsd:choice>
      </xsd:sequence>
      <xsd:attribute name="Id" type="xsd:ID" use="optional" />
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CertTypeType">
  <xsd:attribute name="Type" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="advanced" />
        <xsd:enumeration value="qualified" />
        <xsd:enumeration value="unknown" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="acknowledgementEntryType">
  <xsd:sequence>
    <xsd:element name="Code" type="osci:Number" />
    <xsd:element name="Text" type="xsd:string" />
  </xsd:sequence>
  <xsd:attribute ref="xml:lang" use="optional" />
</xsd:complexType>

<xsd:complexType name="acknowledgementType">
  <xsd:sequence>
    <xsd:element name="Entry" type="osci:acknowledgementEntryType"
     maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="InspectionType">
  <xsd:sequence>
    <xsd:element name="Timestamp" type="osci:TimestampType" />
    <xsd:element name="X509IssuerName" type="xsd:string" />
    <xsd:element name="X509SerialNumber" type="xsd:integer" />
    <xsd:element name="CertType" type="osci:CertTypeType" />
    <xsd:element name="MathResult" type="osci:MathResultType" />
    <xsd:element name="OfflineResult" type="osci:OfflineResultType" />
    <xsd:element name="OnlineResult" type="osci:OnlineResultType"
     minOccurs="0" />
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="InspectionReportType">
  <xsd:sequence>
    <xsd:element name="Inspection" type="osci:InspectionType"
```

```
      minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="MathResultType">
  <xsd:attribute name="Result" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="ok" />
        <xsd:enumeration value="corrupted" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="OfflineResultType">
  <xsd:attribute name="Result" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="valid" />
        <xsd:enumeration value="invalid" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="OnlineResultType">
  <xsd:choice>
    <xsd:element name="OSCP" type="xsd:base64Binary" />
    <xsd:element name="CRL" type="xsd:dateTime" />
    <xsd:element name="LDAP">
      <xsd:complexType />
    </xsd:element>
  </xsd:choice>
  <xsd:attribute name="Result" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="ok" />
        <xsd:enumeration value="revoked" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>

<xsd:complexType name="TimestampType">
  <xsd:choice>
    <xsd:element name="Plain" type="xsd:dateTime" />
    <xsd:element name="Cryptographic">
      <xsd:complexType>
        <xsd:simpleContent>
```

```
          <xsd:extension base="xsd:base64Binary">
            <xsd:attribute name="Algorith" type="xsd:anyURI" />
          </xsd:extension>
        </xsd:simpleContent>
      </xsd:complexType>
    </xsd:element>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="ContentType">
  <xsd:sequence>
    <xsd:any namespace="##any" minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
  <xsd:anyAttribute namespace="##any" />
</xsd:complexType>

<xsd:complexType name="ContentContainerType">
  <xsd:sequence>
    <xsd:element ref="ds:Signature"
     minOccurs="0" maxOccurs="unbounded" />
    <xsd:element name="Content" type="osci:ContentType" minOccurs="0" />
    <xsd:element ref="xenc:EncryptedData"
     minOccurs="0" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

  <!-- ### common templates ### -->

<xsd:complexType name="ProcessCardBundleTemplate" abstract="true">
  <xsd:sequence>
    <xsd:element name="MessageId" type="osci:MessageIdType" />
    <xsd:element name="ProcessCard" type="osci:ProcessCardTemplate" />
    <xsd:element name="InspectionReport"
     type="osci:InspectionReportType" />
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ProcessCardTemplate" abstract="true">
  <xsd:sequence>
    <xsd:element name="Creation" type="osci:TimestampType"
     minOccurs="0" />
    <xsd:element name="Forwarding" type="osci:TimestampType"
     minOccurs="0" />
    <xsd:element name="Reception" type="osci:TimestampType"
     minOccurs="0" />
    <xsd:element name="Subject" type="xsd:string"
     minOccurs="0" />
  </xsd:sequence>
  <xsd:attribute name="RecentModification" type="xsd:dateTime"
   use="required"/>
```

```
</xsd:complexType>

  <!-- ### SOAP header and body block types ### -->

<xsd:complexType name="SignatureBlockType">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultHeaderBlockTemplate">
      <xsd:sequence>
        <xsd:element ref="ds:Signature" />
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="DesiredLanguagesType">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultHeaderBlockTemplate">
      <xsd:attribute name="LanguagesList" use="required">
        <xsd:simpleType>
          <xsd:restriction base="osci:LanguagesListType">
            <xsd:minLength value="1" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="QualityOfTimestampType">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultHeaderBlockTemplate">
      <xsd:attribute name="Service" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="creation" />
            <xsd:enumeration value="reception" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="Quality" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="plain" />
            <xsd:enumeration value="cryptographic" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

```
<xsd:complexType name="ContentPackageType">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultBodyBlockTemplate">
      <xsd:sequence>
        <xsd:element ref="xenc:EncryptedData"
         minOccurs="0" maxOccurs="unbounded" />
        <xsd:element name="ContentContainer"
         type="osci:ContentContainerType"
         minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

  <!-- ### SOAP header and body templates ### -->

<xsd:complexType name="GeneralHeaderBlockTemplate" abstract="true">
  <xsd:attribute name="Id" type="xsd:ID" use="required" />
  <xsd:attribute ref="soap:mustUnderstand" fixed="1" use="required" />
</xsd:complexType>

<xsd:complexType name="DefaultHeaderBlockTemplate" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="osci:GeneralHeaderBlockTemplate">
      <xsd:attribute ref="soap:actor"
       fixed="http://schemas.xmlsoap.org/soap/actor/next"
       use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="GeneralBodyBlockTemplate" abstract="true">
</xsd:complexType>

<xsd:complexType name="DefaultBodyBlockTemplate" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultBodyBlockTemplate">
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ControlBlockTemplate" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultHeaderBlockTemplate">
      <xsd:sequence>
        <xsd:element name="Response" type="xsd:string" minOccurs="0" />
        <xsd:element name="Challenge" type="xsd:string" minOccurs="0" />
      </xsd:sequence>
      <xsd:attribute name="ConversationId" type="osci:Number"
```

```
        use="optional" />
      <xsd:attribute name="SequenceNumber" type="osci:Number"
        use="optional" />
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="IntermediaryCertificatesTemplate" abstract="true">
  <xsd:complexContent>
    <xsd:extension base="osci:GeneralHeaderBlockTemplate">
      <xsd:sequence>
        <xsd:element name="CipherCertificateIntermediary"
          type="osci:CertificateType" minOccurs="0" />
        <xsd:element name="SignatureCertificateIntermediary"
          type="osci:CertificateType" minOccurs="0" />
      </xsd:sequence>
      <xsd:attribute ref="soap:actor"
        fixed="http://www.w3.org/2001/12/soap-envelope/actor/none"
        use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="NonIntermediaryCertificatesTemplate"
 abstract="true">
  <xsd:complexContent>
    <xsd:extension base="osci:GeneralHeaderBlockTemplate">
      <xsd:sequence>
        <xsd:element name="CipherCertificateOriginator"
          type="osci:CertificateType" minOccurs="0" />
        <xsd:element name="CipherCertificateOtherAuthor"
          type="osci:CertificateType"
          minOccurs="0" maxOccurs="unbounded" />
        <xsd:element name="CipherCertificateAddressee"
          type="osci:CertificateType" minOccurs="0" />
        <xsd:element name="CipherCertificateOtherReader"
          type="osci:CertificateType"
          minOccurs="0" maxOccurs="unbounded" />
        <xsd:element name="SignatureCertificateOriginator"
          type="osci:CertificateType" minOccurs="0" />
        <xsd:element name="SignatureCertificateOtherAuthor"
          type="osci:CertificateType"
          minOccurs="0" maxOccurs="unbounded" />
      </xsd:sequence>
      <xsd:attribute ref="soap:actor"
        fixed="http://www.w3.org/2001/12/soap-envelope/actor/none"
        use="required"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

```
</xsd:schema>
```

6.4 Adaption of XML Signature

For the use of XML Signature in OSCI Transport the following restrictions are valid:

```
<xsd:schema targetNamespace="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.w3.org/2001/04/xmlenc#"
   schemaLocation="oscienc.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - restriction on XML Signature
      $RCSfile: oscisig.xsd,v $, $Revision: 1.7 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### redefinitions ### -->

  <xsd:redefine schemaLocation="http://www.w3.org/TR/2001/CR-xmldsig-core-
20010419/xmldsig-core-schema.xsd">

    <xsd:complexType name="KeyInfoType">
      <xsd:complexContent>
        <xsd:restriction base="ds:KeyInfoType">
          <xsd:choice>
            <xsd:element ref="xenc:EncryptedKey" />
            <xsd:element ref="ds:RetrievalMethod" />
            <xsd:element ref="ds:X509Data" />
          </xsd:choice>
          <xsd:attribute name="Id" type="xsd:ID" use="optional" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="ReferenceType">
      <xsd:complexContent>
        <xsd:restriction base="ds:ReferenceType">
          <xsd:sequence>
            <xsd:element ref="ds:Transforms" />
            <xsd:element ref="ds:DigestMethod" />
            <xsd:element ref="ds:DigestValue" />
          </xsd:sequence>
```

```
        <xsd:attribute name="URI" type="xsd:anyURI" use="optional" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="SignatureType">
    <xsd:complexContent>
      <xsd:restriction base="ds:SignatureType">
        <xsd:sequence>
          <xsd:element ref="ds:SignedInfo" />
          <xsd:element ref="ds:SignatureValue" />
          <xsd:element ref="ds:KeyInfo" />
          <xsd:element ref="ds:Object"
           minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="SignatureValueType">
    <xsd:simpleContent>
      <xsd:restriction base="ds:SignatureValueType">
        <xsd:attribute name="Id" type="xsd:ID" use="optional" />
      </xsd:restriction>
    </xsd:simpleContent>
  </xsd:complexType>

  <xsd:complexType name="RetrievalMethodType">
    <xsd:complexContent>
      <xsd:restriction base="ds:RetrievalMethodType">
        <xsd:attribute name="URI" type="xsd:anyURI" use="required" />
        <xsd:attribute name="Type">
          <xsd:simpleType>
            <xsd:restriction base="xsd:anyURI">
              <xsd:enumeration
               value="http://www.w3.org/2000/09/xmldsig#X509Data" />
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="X509DataType">
    <xsd:complexContent>
      <xsd:restriction base="ds:X509DataType">
        <xsd:sequence maxOccurs="1">
          <xsd:element name="X509Certificate" type="xsd:base64Binary" />
        </xsd:sequence>
      </xsd:restriction>
```

```
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="CanonicalizationMethodType">
    <xsd:complexContent>
      <xsd:restriction base="ds:CanonicalizationMethodType">
        <xsd:attribute name="Algorithm" use="required">
          <xsd:simpleType>
            <xsd:restriction base="xsd:anyURI">
              <xsd:enumeration
               value="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="DigestMethodType">
    <xsd:complexContent>
      <xsd:restriction base="ds:DigestMethodType">
        <xsd:attribute name="Algorithm" use="required">
          <xsd:simpleType>
            <xsd:restriction base="xsd:anyURI">
              <xsd:enumeration
               value="http://www.w3.org/2000/09/xmldsig#sha1" />
              <xsd:enumeration
               value="http://www.osci.de/2002/04/osci#ripemd160" />
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="SignatureMethodType">
    <xsd:complexContent>
      <xsd:restriction base="ds:SignatureMethodType">
        <xsd:attribute name="Algorithm" use="required">
          <xsd:simpleType>
            <xsd:restriction base="xsd:anyURI">
              <xsd:enumeration
               value="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
              <xsd:enumeration
               value="http://www.osci.de/2002/04/osci#rsa-ripemd160" />
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:restriction>
    </xsd:complexContent>
```

```
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

This schema allows especially to use a signature of type XAdES (cf. [XADES]) as signature for content data.

On the business transaction level the signatures in the elements `ds:Signature` can refer to any elements of the enclosing content data container.

The signature of a request refers to all the elements that are lying parallel to `osci:ClientSignature` in the `soap:Header`, as well as to the element `soap:Body`.

The signature of a response refers to all the elements that are lying parallel to `osci:SupplierSignature` in the `soap:Header`, as well as to the element `soap:Body`.

The element `ds:KeyInfo` within the element `ds:Signature` refers to a certificate within the same request or within the same response.

6.5 Adaption of XML Encryption

For use of XML Encryption in OSCI Transport the following restrictions are valid:

```
<xsd:schema targetNamespace="http://www.w3.org/2001/04/xmlenc#"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            elementFormDefault="qualified">

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - restriction on XML Encryption
      $RCSfile: oscienc.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### redefinitions ### -->

  <xsd:redefine
   schemaLocation="http://www.w3.org/TR/xmlenc-core/xenc-schema.xsd">

    <xsd:complexType name="EncryptionMethodType">
      <xsd:complexContent>
        <xsd:restriction base="xenc:EncryptionMethodType">
          <xsd:sequence>
            <xsd:element name="KeySize" minOccurs="0"
             type="xenc:KeySizeType" />
          </xsd:sequence>
          <xsd:attribute name="Algorithm" use="required">
            <xsd:simpleType>
```

```
              <xsd:restriction base="xsd:anyURI">
                <xsd:enumeration
                 value="http://www.w3.org/2001/04/xmlenc#tripledes-cbc" />
                <xsd:enumeration
                 value="http://www.w3.org/2001/04/xmlenc#aes128-cbc" />
                <xsd:enumeration
                 value="http://www.w3.org/2001/04/xmlenc#aes192-cbc" />
                <xsd:enumeration
                 value="http://www.w3.org/2001/04/xmlenc#aes256-cbc" />
                <xsd:enumeration
                 value="http://www.w3.org/2001/04/xmlenc#rsa-1_5" />
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:attribute>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="CipherReferenceType">
      <xsd:complexContent>
        <xsd:restriction base="xenc:CipherReferenceType">
          <xsd:attribute name="URI" type="xsd:anyURI" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="EncryptedDataType">
      <xsd:complexContent>
        <xsd:restriction base="xenc:EncryptedDataType">
          <xsd:sequence>
            <xsd:element name="EncryptionMethod"
             type="xenc:EncryptionMethodType" minOccurs="0" />
            <xsd:element ref="ds:KeyInfo" minOccurs="0" />
            <xsd:element ref="xenc:CipherData" minOccurs="1" />
          </xsd:sequence>
          <xsd:attribute name="MimeType" type="xsd:string" use="optional"
/>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="EncryptedKeyType">
      <xsd:complexContent>
        <xsd:restriction base="xenc:EncryptedKeyType">
          <xsd:sequence>
            <xsd:element name="EncryptionMethod"
             type="xenc:EncryptionMethodType" minOccurs="1" />
            <xsd:element ref="ds:KeyInfo" minOccurs="1" />
            <xsd:element ref="xenc:CipherData" minOccurs="1" />
          </xsd:sequence>
```

```
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

In the encryption of content data, the encryption information in the elements `xenc:EncryptedData` can refer to any content data containers and attachments that are in the same delivery as the element `xenc:EncryptedData`.

6.6 Requests and Responses

In OSCI Transport clients can send requests of the following types:

- *Dialog init request*: A client opens an explicit dialog with a supplier.
- *Dialog exit request*: A client closes an explicit dialog with a supplier.
- *MessageId request*: A user orders a MessageId at the intermediary.
- *Delivery request*: A user sends a delivery to an intermediary, so that the latter will have it ready for collection by another user.
- *Fetch delivery request*: A user collects a delivery at an intermediary's that another user has handed in before by means of a delivery request.
- *Fetch process card request*: A user collects one or several process cards at an intermediary's. In a process card the way of a delivery from a sender to a recipient is recorded.
- *Forwarding delivery request*: A user sends a delivery to an intermediary, so that the latter will forward it to a service provider.
- *Acceptance request*: An intermediary sends a delivery to a service provider.
- *Mediate delivery request*: A user sends a delivery to an intermediary, so that the latter will forward it to a service provider. The user expects in the intermediary's response a delivery of the service provider.
- *Processing request*: An intermediary asks a service provider to handle a delivery of a user and to supply another delivery as a reaction.

A supplier reacts to each request of a client with the corresponding response type.

6.6.1 Dialog Init Request

A dialog init request is realised as SOAP envelope.

The following example explains the structure of a dialog init request:

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

<!-- XML-Beispiel OSCI 1.2 Dialoginitialisierungsauftrag -->
<!-- $RCSfile: InitDialog.xml,v $, $Revision: 1.8 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
```

```
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapInitDialog.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:Challenge>AAQCAEMmCZtuMFQxDSa</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
    <osci:NonIntermediaryCertificates Id="XREF-0010"
     soap:mustUnderstand="1"
     soap:actor="http://www.w3.org/2001/12/soap-envelope/actor/none">
      <osci:CipherCertificateOriginator>
        <ds:X509Data>
          <ds:X509Certificate>
            Chiffrierzertifikat/Benutzer/Base64codiert
          </ds:X509Certificate>
        </ds:X509Data>
      </osci:CipherCertificateOriginator>
    </osci:NonIntermediaryCertificates>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:initDialog />
  </soap:Body>
</soap:Envelope>
```

A dialog init request conforms to the following schemas.

- Adaption of a SOAP envelope for dialog init requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="InitDialog.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
```

```
      OSCI 1.2 - dialog init request SOAP-Envelope
      $RCSfile: soapInitDialog.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:NonIntermediaryCertificates" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:initDialog" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>
```

```
</xsd:schema>
```

- Schema for dialog init requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - dialog init request
      $RCSfile: InitDialog.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="prohibited" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="prohibited" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:NonIntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="CipherCertificateOriginator"
           type="osci:CertificateType" />
          <xsd:element name="SignatureCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
```

```
  </xsd:complexType>

  <xsd:complexType name="initDialogType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="initDialog" type="osci:initDialogType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.2 Dialog Init Response

A dialog init response is realised as SOAP envelope.

The following example explains the structure of a dialog init response:

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

<!-- XML-Beispiel OSCI 1.2 Dialoginitialisierungsantwort -->
<!-- $RCSfile: ResponseToInitDialog.xml,v $, $Revision: 1.8 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToInitDialog.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
```

```
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123">
      <osci:Response>AAQCAEMmCZtuMFQxDSa</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:responseToInitDialog>
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0801</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog weiterhin
geöffnet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
    </osci:responseToInitDialog>
  </soap:Body>
</soap:Envelope>
```

A dialog init response conforms to the following schemas.

- Adaption of a SOAP envelope for dialog init responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToInitDialog.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - dialog init response SOAP-Envelope
      $RCSfile: soapResponseToInitDialog.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
```

```
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:responseToInitDialog" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for dialog init responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
```

```
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - dialog init response
      $RCSfile: ResponseToInitDialog.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required"/>
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="prohibited" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="IntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToInitDialogType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

```
  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="responseToInitDialog"
   type="osci:responseToInitDialogType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.3 Dialog Exit Request

A dialog exit request is realised as SOAP envelope.

The following example explains the structure of a dialog exit request:

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

<!-- XML-Beispiel OSCI 1.2 Dialogendeauftrag  -->
<!-- $RCSfile: ExitDialog.xml,v $, $Revision: 1.9 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapExitDialog.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="5">
      <osci:Response>AAQCAEMmCZtuMFQxDSa</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
  </soap:Header>
```

```
  <soap:Body Id="XREF-0100">
    <osci:exitDialog />
  </soap:Body>
</soap:Envelope>
```

A dialog exit request conforms to the following schemas.

- Adaption of a SOAP envelope for dialog exit requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ExitDialog.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - dialog exit request SOAP-Envelope
      $RCSfile: soapExitDialog.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
```

```
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:exitDialog" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for dialog exit requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - dialog exit request
      $RCSfile: ExitDialog.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
```

```
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required"/>
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required"/>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

    <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:NonIntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="exitDialogType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="exitDialog" type="osci:exitDialogType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.4 Dialog Exit Response

A dialog exit response is realised as SOAP envelope.

The following example explains the structure of a dialog exit response:

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

<!-- XML-Beispiel OSCI 1.2 Dialogendeantwort -->
<!-- $RCSfile: ResponseToExitDialog.xml,v $, $Revision: 1.8 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToExitDialog.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="6">
      <osci:Response>345nchfcfoqc5dfg</osci:Response>
    </osci:ControlBlock>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:responseToExitDialog>
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0800</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog beendet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
    </osci:responseToExitDialog>
  </soap:Body>
</soap:Envelope>
```

A dialog exit response conforms to the following schemas.

- Adaption of a SOAP envelope for dialog exit responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
```

```
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToExitDialog.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - dialog exit response SOAP-Envelope
      $RCSfile: soapResponseToExitDialog.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
```

```
            <xsd:element ref="osci:responseToExitDialog" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for dialog exit responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - dialog exit response
      $RCSfile: ResponseToExitDialog.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string"
           minOccurs="0" maxOccurs="0" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required"/>
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required"/>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="IntermediaryCertificatesType">
```

```
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToExitDialogType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="responseToExitDialog"
   type="osci:responseToExitDialogType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.5 MessageId Request

A MessageId request is realised as SOAP envelope.

The following example explains the structure of a MessageId request:

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

<!-- XML-Beispiel OSCI 1.2 Message-ID-Anforderungsauftrag  -->
<!-- $RCSfile: GetMessageId.xml,v $, $Revision: 1.9 $ -->
```

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapGetMessageId.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="5">
      <osci:Response>AAQCAEMmCZtuMFQxDSa</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:getMessageId />
  </soap:Body>
</soap:Envelope>
```

A MessageId request conforms to the following schemas.

- Adaption of a SOAP envelope for MessageId call requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="GetMessageId.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - messageId request SOAP-Envelope
      $RCSfile: soapGetMessageId.xsd,v $, $Revision: 1.7 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->
```

```
  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:getMessageId" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for MessageId requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - messageId request
      $RCSfile: GetMessageId.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="0" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="optional" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required"/>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:NonIntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="CipherCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
          <xsd:element name="SignatureCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="getMessageIdType">
    <xsd:complexContent>
```

```
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="getMessageId" type="osci:getMessageIdType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.6 MessageId Response

A MessageId response is realised as SOAP envelope.

The following example explains the structure of a MessageId response:

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

<!-- XML-Beispiel OSCI 1.2 Message-ID-Anforderungsantwort -->
<!-- $RCSfile: ResponseToGetMessageId.xml,v $, $Revision: 1.9 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToGetMessageId.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="6">
      <osci:Response>345nchfcfoqc5dfg</osci:Response>
```

```
      <osci:Challenge>n5vl2dfie9uf6d</osci:Challenge>
    </osci:ControlBlock>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:responseToGetMessageId>
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0801</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog weiterhin
geöffnet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
      <osci:MessageId>MessageId/base64codiert</osci:MessageId>
    </osci:responseToGetMessageId>
  </soap:Body>
</soap:Envelope>
```

A MessageId response conforms to the following schemas.

- Adaption of a SOAP envelope for MessageId responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToGetMessageId.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - messageId response SOAP-Envelope
      $RCSfile: soapResponseToGetMessageId.xsd,v $, $Revision: 1.7 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
```

```
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:responseToGetMessageId" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for MessageId responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />
```

```
  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - messageId response
      $RCSfile: ResponseToGetMessageId.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="IntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToGetMessageIdType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
          <xsd:element name="MessageId" type="osci:MessageIdType"
           minOccurs="0" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->
```

```
  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="responseToGetMessageId"
   type="osci:responseToGetMessageIdType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.7 Delivery Request

A delivery request is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of a delivery request:

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

<!-- XML-Beispiel OSCI 1.2 Zustellungsauftrag  -->
<!-- $RCSfile: StoreDelivery.xml,v $, $Revision: 1.9 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapStoreDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="5">
      <osci:Response>AAQCAEMmCZtuMFQxDSa</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
    <osci:QualityOfTimestamp Id="XREF-0003" soap:mustUnderstand="1"
```

```
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     Service="creation" Quality="plain" />
    <osci:QualityOfTimestamp Id="XREF-0004" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     Service="reception" Quality="plain" />
    <osci:storeDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:MessageId>MessageId/Base64codiert</osci:MessageId>
      <osci:Subject>Betreff der Zustellung</osci:Subject>
    </osci:storeDelivery>
    <osci:NonIntermediaryCertificates Id="XREF-0010"
     soap:mustUnderstand="1"
     soap:actor="http://www.w3.org/2001/12/soap-envelope/actor/none">
      <osci:CipherCertificateAddressee>
        <ds:X509Data>
          <ds:X509Certificate>
            Chiffrierzertifikat/Empfaenger/Base64codiert
          </ds:X509Certificate>
        </ds:X509Data>
      </osci:CipherCertificateAddressee>
    </osci:NonIntermediaryCertificates>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:ContentPackage>
      <osci:ContentContainer>
        <osci:Content />
      </osci:ContentContainer>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

A delivery request (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for delivery responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="StoreDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - delivery request SOAP-Envelope
```

```
      $RCSfile: soapStoreDelivery.xsd,v $, $Revision: 1.6 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:QualityOfTimestamp"
             minOccurs="2" maxOccurs="2" />
            <xsd:element ref="osci:storeDelivery" />
            <xsd:element ref="osci:NonIntermediaryCertificates" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>
```

```
  </xsd:redefine>

</xsd:schema>
```

- Schema for delivery requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - delivery request
      $RCSfile: storeDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="0" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="optional" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="storeDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:extension>
```

```
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:NonIntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="CipherCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
          <xsd:element name="CipherCertificateOtherAuthor"
           type="osci:CertificateType"
           minOccurs="0" maxOccurs="unbounded" />
          <xsd:element name="CipherCertificateAddressee"
           type="osci:CertificateType" minOccurs="1" />
          <xsd:element name="CipherCertificateOtherReader"
           type="osci:CertificateType"
           minOccurs="0" maxOccurs="unbounded" />
          <xsd:element name="SignatureCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
          <xsd:element name="SignatureCertificateOtherAuthor"
           type="osci:CertificateType"
           minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="QualityOfTimestamp"
   type="osci:QualityOfTimestampType" />

  <xsd:element name="storeDelivery" type="osci:storeDeliveryType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)
- Adaption of XML Encryption (cf. Section 6.5)

6.6.8 Delivery Response

A delivery response is realised as SOAP envelope.

The following example explains the structure of a delivery response:

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

<!-- XML-Beispiel OSCI 1.2 Zustellungsantwort -->
<!-- $RCSfile: ResponseToStoreDelivery.xml,v $, $Revision: 1.8 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToStoreDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="6">
      <osci:Response>345nchfcfoqc5dfg</osci:Response>
      <osci:Challenge>n5vl2dfie9uf6d</osci:Challenge>
    </osci:ControlBlock>
    <osci:responseToStoreDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0801</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog weiterhin
geöffnet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
      <osci:ProcessCardBundle>
        <osci:MessageId>MessageId/Base64codiert</osci:MessageId>
        <osci:ProcessCard
         RecentModification="2002-04-23T23:41:05.527-01:00">
          <osci:Creation>
            <osci:Plain>2002-04-23T23:41:05.527-01:00</osci:Plain>
          </osci:Creation>
          <osci:Subject>Betreff der Zustellung</osci:Subject>
        </osci:ProcessCard>
        <osci:InspectionReport />
      </osci:ProcessCardBundle>
```

```
    </osci:responseToStoreDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100" />
</soap:Envelope>
```

If the intermediary is able to create a process card for the delivery handed in by means of a *delivery request*, then the process card will be sent to the user within the *delivery response.*

A delivery response conforms to the following schemas.

- Adaption of a SOAP envelope for delivery responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToStoreDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - delivery response SOAP-Envelope
      $RCSfile: soapResponseToStoreDelivery.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
```

```
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:responseToStoreDelivery" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for delivery responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - delivery response
      $RCSfile: ResponseToStoreDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
```

```
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="ProcessCardBundleType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardBundleTemplate">
        <xsd:sequence>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="ProcessCard" type="osci:ProcessCardType" />
          <xsd:element name="InspectionReport"
           type="osci:InspectionReportType" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="ProcessCardType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardTemplate">
        <xsd:sequence>
          <xsd:element name="Creation" type="osci:TimestampType" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToStoreDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
          <xsd:element name="ProcessCardBundle"
           type="osci:ProcessCardBundleType" minOccurs="0" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

```
  <xsd:complexType name="IntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="responseToStoreDelivery"
   type="osci:responseToStoreDeliveryType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.9 Fetch Delivery Request

A fetch delivery request is realised as SOAP envelope.

The following example explains the structure of a fetch delivery request:

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

<!-- XML-Beispiel OSCI 1.2 Zustellungsabholauftrag  -->
<!-- $RCSfile: FetchDelivery.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapFetchDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
```

```
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="5">
      <osci:Response>AAQCAEMmCZtuMFQxDSa</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
    <osci:fetchDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:SelectionRule>
        <osci:ReceptionOfDelivery>2002-04-01T12:00:00.000-
01:00</osci:ReceptionOfDelivery>
      </osci:SelectionRule>
    </osci:fetchDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100" />
</soap:Envelope>
```

By means of the element `osci:SelectionRule` the user has the option to determine selection criteria, with which an ordered delivery can be specified more precisely. As selection criteria the user can determine either a time of reception or a MessageId as desired.

The intermediary determines a delivery that is to be sent to the user in a *fetch delivery response* on the basis of the following rules:

1. If the element `osci:MessageId` is present, the delivery is returned that corresponds to the MessageId stated in `osci:MessageId`.
2. If the element `osci:ReceptionOfDelivery` is present, the oldest delivery (time of presentation at the intermediary's) is returned whose time of presentation at the intermediary's is after the time of reception stated in `osci:ReceptionOfDelivery`.
3. If neither the element `osci:ReceptionOfDelivery` nor the element `osci:MessageId` is present, the delivery with the oldest time of presentation at the intermediary's is returned.

A fetch delivery request conforms to the following schemas.

- Adaption of a SOAP envelope for fetch delivery requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
```

```
   schemaLocation="FetchDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - fetch delivery request SOAP-Envelope
      $RCSfile: soapFetchDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:fetchDelivery" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
```

```
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for fetch delivery requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - fetch delivery request
      $RCSfile: fetchDelivery.xsd,v $, $Revision: 1.2 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="fetchDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="SelectionRule" minOccurs="0">
            <xsd:complexType>
              <xsd:choice>
```

```
                <xsd:element name="ReceptionOfDelivery"
                 type="xsd:dateTime" />
                <xsd:element name="MessageId" type="osci:MessageIdType" />
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:NonIntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="fetchDelivery" type="osci:fetchDeliveryType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.10 Fetch Delivery Response

A fetch delivery response is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of a fetch delivery response:

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

```
<!-- XML-Beispiel OSCI 1.2 Zustellungsabholantwort -->
<!-- $RCSfile: ResponseToFetchDelivery.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToFetchDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="6">
      <osci:Response>n5vl2dfie9uf6d</osci:Response>
      <osci:Challenge>fie9uf6dGGFRGm3</osci:Challenge>
    </osci:ControlBlock>
    <osci:responseToFetchDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0801</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog weiterhin
geöffnet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
      <osci:fetchDelivery>
        <osci:SelectionRule>
          <osci:ReceptionOfDelivery>2002-04-01T12:00:00.000-
01:00</osci:ReceptionOfDelivery>
        </osci:SelectionRule>
      </osci:fetchDelivery>
      <osci:ProcessCardBundle>
        <osci:MessageId>MessageId/Base64codiert</osci:MessageId>
        <osci:ProcessCard RecentModification="2002-04-24T12:15:01.516-
01:00">
          <osci:Creation>
            <osci:Plain>2002-04-23T23:41:05.527-01:00</osci:Plain>
          </osci:Creation>
          <osci:Forwarding>
            <osci:Plain>2002-04-24T12:15:01.516-01:00</osci:Plain>
          </osci:Forwarding>
          <osci:Subject>Betreff der Zustellung</osci:Subject>
        </osci:ProcessCard>
        <osci:InspectionReport>
          <osci:Inspection>
            <osci:Timestamp>
```

```
              <osci:Plain>2002-04-24T12:14:32.291-01:00</osci:Plain>
            </osci:Timestamp>
            <osci:X509IssuerName>Otto Mustermann</osci:X509IssuerName>
            <osci:X509SerialNumber>1625843</osci:X509SerialNumber>
            <osci:CertType Type="qualified" />
            <osci:MathResult Result="ok" />
            <osci:OfflineResult Result="valid" />
            <osci:OnlineResult Result="ok">
              <osci:LDAP />
            </osci:OnlineResult>
          </osci:Inspection>
        </osci:InspectionReport>
      </osci:ProcessCardBundle>
    </osci:responseToFetchDelivery>
    <osci:NonIntermediaryCertificates Id="XREF-0020"
     soap:mustUnderstand="1"
     soap:actor="http://www.w3.org/2001/12/soap-envelope/actor/none">
      <osci:CipherCertificateAddressee Id="XREF-4444">
        <ds:X509Data>
          <ds:X509Certificate>
            Chiffrierzertifikat/Empfaenger/Base64codiert
          </ds:X509Certificate>
        </ds:X509Data>
      </osci:CipherCertificateAddressee>
    </osci:NonIntermediaryCertificates>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:ContentPackage>
      <xenc:EncryptedData>
        <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc" />
        <ds:KeyInfo>
          <xenc:EncryptedKey>
            <xenc:EncryptionMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5">
              <xenc:KeySize>1024</xenc:KeySize>
            </xenc:EncryptionMethod>
            <ds:KeyInfo>
              <ds:RetrievalMethod URI="#XREF-4444" />
            </ds:KeyInfo>
            <xenc:CipherData>

<xenc:CipherValue>Verschluesselter/Containerschluessel/Base64codiert</xenc:
CipherValue>
            </xenc:CipherData>
          </xenc:EncryptedKey>
        </ds:KeyInfo>
        <xenc:CipherData>
```

```
<xenc:CipherValue>Verschluesselter/Inhaltsdatencontainer/Base64codiert</xen
c:CipherValue>
        </xenc:CipherData>
      </xenc:EncryptedData>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

The interpretation of the selection criteria in the element `SelectionRule` of a *fetch delivery request* is described in Section 6.6.9.

In addition to the rules stated in Section 6.6.9 the following applies:

- If at the intermediary's (according to points 1.-3. of the rules stated in 6.6.9) there are no deliveries for the user, no delivery will be sent along with the *fetch delivery response* and the user will receive a corresponding acknowledgement (cf. Chapter 5).
- If the intermediary has ready further deliveries for the user, the user will receive an according acknowledgement (cf. Chapter 5). The presence of further deliveries does not depend on the selection criteria listed in the element `osci:SelectionRule` of the *fetch delivery request*.

A fetch delivery response (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for fetch delivery responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToFetchDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - fetch delivery response SOAP-Envelope
      $RCSfile: soapResponseToFetchDelivery.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->
```

```
    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:responseToFetchDelivery" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" minOccurs="0" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for fetch delivery responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
```

```
          xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
          xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
          xmlns:osci="http://www.osci.de/2002/04/osci"
          attributeFormDefault="unqualified"
          elementFormDefault="qualified">

<xsd:include schemaLocation="order.xsd" />

<xsd:annotation>
  <xsd:documentation xml:lang="de">
    OSCI 1.2 - fetch delivery response
    $RCSfile: ResponseToFetchDelivery.xsd,v $, $Revision: 1.3 $
  </xsd:documentation>
</xsd:annotation>

<!-- ### derived types ### -->

<xsd:complexType name="ControlBlockType">
  <xsd:complexContent>
    <xsd:restriction base="osci:ControlBlockTemplate">
      <xsd:sequence>
        <xsd:element name="Response" type="xsd:string" minOccurs="1" />
        <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
      </xsd:sequence>
      <xsd:attribute name="ConversationId" type="osci:Number"
       use="required" />
      <xsd:attribute name="SequenceNumber" type="osci:Number"
       use="required" />
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ProcessCardBundleType">
  <xsd:complexContent>
    <xsd:restriction base="osci:ProcessCardBundleTemplate">
      <xsd:sequence>
        <xsd:element name="MessageId" type="osci:MessageIdType" />
        <xsd:element name="ProcessCard" type="osci:ProcessCardType" />
        <xsd:element name="InspectionReport"
         type="osci:InspectionReportType" />
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ProcessCardType">
  <xsd:complexContent>
    <xsd:restriction base="osci:ProcessCardTemplate">
      <xsd:sequence>
        <xsd:element name="Creation" type="osci:TimestampType" />
```

```
          <xsd:element name="Forwarding" type="osci:TimestampType" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToFetchDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
          <xsd:element name="fetchDelivery">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:any namespace="http://www.osci.de/2002/04/osci"
                 maxOccurs="unbounded" processContents="strict"/>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="ProcessCardBundle"
           type="osci:ProcessCardBundleType" minOccurs="0" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="IntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:extension base="osci:NonIntermediaryCertificatesTemplate" />
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />
```

```
  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="responseToFetchDelivery"
   type="osci:responseToFetchDeliveryType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)
- Adaption of XML Encryption (cf. Section 6.5)

6.6.11 Fetch Process Card Request

A fetch process card request is realised as SOAP envelope.

The following example explains the structure of a fetch process card request:

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

<!-- XML-Beispiel OSCI 1.2 Laufzettelabholauftrag  -->
<!-- $RCSfile: FetchProcessCard.xml,v $, $Revision: 1.6 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapFetchProcessCard.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="5">
      <osci:Response>AAQCAEMmCZtuMFQxDSa</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
```

```
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:fetchProcessCard>
      <osci:SelectionRule>
        <osci:RecentModification>2002-04-01T12:00:00.000-
01:00</osci:RecentModification>
      </osci:SelectionRule>
      <osci:Quantity Limit="1" />
    </osci:fetchProcessCard>
  </soap:Body>
</soap:Envelope>
```

In a *fetch process card request* the element `osci:SelectionRule` is used to specify more precisely the ordered process cards.

The user can state a time of reception, a time of modification or an unlimited number of MessageIds as desired.

The intermediary determines the process cards that are to be sent to the user in a *fetch process card response* on the basis of the following rules:

1. To each element `osci:MessageId` present, the process card is returned that corresponds to the MessageId stated in `osci:MessageId`.
2. If the element `osci:ReceptionOfDelivery` is present, all the process cards for deliveries are returned whose time of presentation at the intermediary's is more recent than the time of reception indicated in `osci:ReceptionOfDelivery`.
3. If the element `osci:RecentModification` is present, all the process cards are returned whose latest time of modification is more recent than the time indicated in `osci:RecentModification`.
4. If none of the elements `osci:ReceptionOfDelivery`, `osci:RecentModification` or `osci:MessageId` has been stated, all the process cards existing for the user are sent.

Furthermore, the user has the option to limit the number of process cards to be returned.

5. If the user has stated an element `osci:Quantity`, from the process cards determined according to points 1.-4. the maximum number stated in the attribute `Limit` will be sent to the user within the *fetch process card response*.

A fetch process card request conforms to the following schemas.

- Adaption of a SOAP envelope for fetch process card requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="FetchProcessCard.xsd" />
```

```
  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - fetch process card request SOAP-Envelope
      $RCSfile: soapFetchProcessCard.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:fetchProcessCard" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
```

```
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for fetch process card requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - fetch process card request
      $RCSfile: FetchProcessCard.xsd,v $, $Revision: 1.3 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:NonIntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateOriginator"
           type="osci:CertificateType" minOccurs="0" />
```

```
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="fetchProcessCardType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
        <xsd:sequence>
          <xsd:element name="SelectionRule" minOccurs="0">
            <xsd:complexType>
              <xsd:choice>
                <xsd:element name="ReceptionOfDelivery"
                 type="xsd:dateTime" />
                <xsd:element name="RecentModification"
                 type="xsd:dateTime" />
                <xsd:element name="MessageId" type="osci:MessageIdType"
                 maxOccurs="unbounded" />
              </xsd:choice>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="Quantity" minOccurs="0">
            <xsd:complexType>
              <xsd:attribute name="Limit" type="xsd:positiveInteger"
               use="required" />
            </xsd:complexType>
          </xsd:element>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="fetchProcessCard" type="osci:fetchProcessCardType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.12 Fetch Process Card Response

A fetch process card response is realised as SOAP envelope.

The following example explains the structure of a fetch process card response:

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

<!-- XML-Beispiel OSCI 1.2 Laufzettelabholantwort -->
<!-- $RCSfile: ResponseToFetchProcessCard.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToFetchProcessCard.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="6">
      <osci:Response>n5vl2dfie9uf6d</osci:Response>
      <osci:Challenge>fie9uf6dGGFRGm3</osci:Challenge>
    </osci:ControlBlock>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:responseToFetchProcessCard>
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>3801</osci:Code>
          <osci:Text>Zu den im Auftrag angegebenen Kriterien liegen weitere
Laufzettel vor</osci:Text>
        </osci:Entry>
        <osci:Entry xml:lang="de">
          <osci:Code>0801</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog weiterhin
geöffnet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
      <osci:fetchProcessCard>
        <osci:SelectionRule>
          <osci:RecentModification>2002-04-01T12:00:00.000-
01:00</osci:RecentModification>
        </osci:SelectionRule>
        <osci:Quantity Limit="1" />
```

```
      </osci:fetchProcessCard>
      <osci:ProcessCardBundle>
        <osci:MessageId>MessageId/1/Base64codiert</osci:MessageId>
        <osci:ProcessCard
         RecentModification="2002-04-24T12:15:01.516-01:00">
          <osci:Creation>
            <osci:Plain>2002-04-23T23:41:05.527-01:00</osci:Plain>
          </osci:Creation>
          <osci:Forwarding>
            <osci:Plain>2002-04-24T12:15:01.516-01:00</osci:Plain>
          </osci:Forwarding>
          <osci:Reception>
            <osci:Plain>2002-04-24T12:17:12.169-01:00</osci:Plain>
          </osci:Reception>
          <osci:Subject>Betreff der Zustellung</osci:Subject>
        </osci:ProcessCard>
        <osci:InspectionReport />
      </osci:ProcessCardBundle>
    </osci:responseToFetchProcessCard>
  </soap:Body>
</soap:Envelope>
```

The interpretation of the selection criteria in the element `SelectionRule` of a *fetch process card request* is described in Section 6.6.11.

In addition to the rules stated in Section 6.6.11 the following applies:

- If according to points 1.-4. of the rules stated in 6.6.11 there are no process cards, this will be communicated to the user in an acknowledgement.
- If according to points 1.-5. there are more process cards than indicated in `Limit`, this will be communicated to the user in an acknowledgement.

A fetch process card response conforms to the following schemas.

- Adaption of a SOAP envelope for fetch process card responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToFetchProcessCard.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - fetch process card response SOAP-Envelope
      $RCSfile: soapResponseToFetchProcessCard.xsd,v $, $Revision: 1.4 $
```

```
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:responseToFetchProcessCard" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for fetch process card responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - fetch process card response
      $RCSfile: ResponseToFetchProcessCard.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="ProcessCardBundleType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardBundleTemplate">
        <xsd:sequence>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="ProcessCard" type="osci:ProcessCardType" />
          <xsd:element name="InspectionReport"
           type="osci:InspectionReportType" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>
```

```
  <xsd:complexType name="ProcessCardType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardTemplate">
        <xsd:sequence>
          <xsd:element name="Creation" type="osci:TimestampType" />
          <xsd:element name="Forwarding" type="osci:TimestampType"
           minOccurs="0" />
          <xsd:element name="Reception" type="osci:TimestampType"
           minOccurs="0" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToFetchProcessCardType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultBodyBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
          <xsd:element name="fetchProcessCard">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:any namespace="http://www.osci.de/2002/04/osci"
                 maxOccurs="unbounded" processContents="strict"/>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="ProcessCardBundle"
           type="osci:ProcessCardBundleType"
           minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="IntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->
```

```
  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="responseToFetchProcessCard"
   type="osci:responseToFetchProcessCardType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.13 Forwarding Delivery Request

A forwarding request is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of a forwarding delivery request:

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

<!-- XML-Beispiel OSCI 1.2 Weiterleitungsauftrag  -->
<!-- $RCSfile: ForwardDelivery.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapForwardDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="5">
      <osci:Response>hZGr4fP+IGrt5</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
    <osci:QualityOfTimestamp Id="XREF-0003" soap:mustUnderstand="1"
```

```
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     Service="creation" Quality="plain" />
    <osci:QualityOfTimestamp Id="XREF-0004" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     Service="reception" Quality="plain" />
    <osci:forwardDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:ContentReceiver
       URI="http://www.osci.de/2002/04/osci#deployer" />
      <osci:MessageId>MessageId/base64codiert</osci:MessageId>
      <osci:Subject>Betreff der Zustellung</osci:Subject>
    </osci:forwardDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:ContentPackage>
      <osci:ContentContainer>
        <osci:Content />
      </osci:ContentContainer>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

A forwarding delivery request (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for forwarding delivery requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ForwardDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - forwarding delivery request SOAP-Envelope
      $RCSfile: soapForwardDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">
```

```
    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:QualityOfTimestamp"
             minOccurs="2" maxOccurs="2" />
            <xsd:element ref="osci:forwardDelivery" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for forwarding delivery requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - forwarding delivery request
      $RCSfile: ForwardDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="0" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="optional" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="forwardDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="ContentReceiver">
            <xsd:complexType>
              <xsd:attribute name="URI" type="xsd:anyURI" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

```
  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:extension base="osci:NonIntermediaryCertificatesTemplate" />
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="QualityOfTimestamp"
   type="osci:QualityOfTimestampType" />

  <xsd:element name="forwardDelivery" type="osci:forwardDeliveryType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)
- Adaption of XML Encryption (cf. Section 6.5)

6.6.14 Forwarding Delivery Response

A forwarding delivery response is realised as SOAP envelope.

The following example explains the structure of a forwarding delivery response:

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

<!-- XML-Beispiel OSCI 1.2 Weiterleitungsantwort -->
<!-- $RCSfile: ResponseToForwardDelivery.xml,v $, $Revision: 1.6 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
```

```
                    soapResponseToForwardDelivery.xsd
                    http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                    http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     ConversationId="87634586123" SequenceNumber="6"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:Response>345nchfcfoqc5dfg</osci:Response>
      <osci:Challenge>n5vl2dfie9uf6d</osci:Challenge>
    </osci:ControlBlock>
    <osci:responseToForwardDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0801</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog weiterhin
geöffnet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
      <osci:ProcessCardBundle>
        <osci:MessageId>MessageId/Base64codiert</osci:MessageId>
        <osci:ProcessCard
         RecentModification="2002-04-24T12:15:01.516-01:00">
          <osci:Creation>
            <osci:Plain>2002-04-23T23:41:05.527-01:00</osci:Plain>
          </osci:Creation>
          <osci:Forwarding>
            <osci:Plain>2002-04-24T12:15:01.516-01:00</osci:Plain>
          </osci:Forwarding>
          <osci:Reception>
            <osci:Plain>2002-04-24T12:17:12.169-01:00</osci:Plain>
          </osci:Reception>
          <osci:Subject>Betreff der Zustellung</osci:Subject>
        </osci:ProcessCard>
        <osci:InspectionReport />
      </osci:ProcessCardBundle>
    </osci:responseToForwardDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100" />
</soap:Envelope>
```

If the intermediary is able to create a process card for the delivery handed in using a *forwarding delivery request*, this process card will be sent to the user within the *forwarding delivery response*.

A forwarding delivery response conforms to the following schemas.

- Adaption of a SOAP envelope for forwarding delivery responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

```
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToForwardDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - forwarding delivery response SOAP-Envelope
      $RCSfile: soapResponseToForwardDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:responseToForwardDelivery" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>
```

```
    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for forwarding delivery responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - forwarding delivery response
      $RCSfile: ResponseToForwardDelivery.xsd,v $, $Revision: 1.3 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>
```

```
  <xsd:complexType name="ProcessCardBundleType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardBundleTemplate">
        <xsd:sequence>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="ProcessCard" type="osci:ProcessCardType" />
          <xsd:element name="InspectionReport"
           type="osci:InspectionReportType" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="ProcessCardType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardTemplate">
        <xsd:sequence>
          <xsd:element name="Creation" type="osci:TimestampType" />
          <xsd:element name="Forwarding"
           type="osci:TimestampType" minOccurs="0" />
          <xsd:element name="Reception" type="osci:TimestampType"
           minOccurs="0" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToForwardDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
          <xsd:element name="ProcessCardBundle"
           type="osci:ProcessCardBundleType" minOccurs="0" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="IntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
```

```
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="responseToForwardDelivery"
   type="osci:responseToForwardDeliveryType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.15 Acceptance Request

An acceptance request is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of an acceptance request:

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

<!-- XML-Beispiel OSCI 1.2 Annahmeauftrag  -->
<!-- $RCSfile: AcceptDelivery.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapAcceptDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:Challenge>xaLW8jIJcCAwEA</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
```

```
    <osci:acceptDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:ProcessCardBundle>
        <osci:MessageId>MessageId/Base64codiert</osci:MessageId>
        <osci:ProcessCard
         RecentModification="2002-04-24T12:15:01.516-01:00">
          <osci:Creation>
            <osci:Plain>2002-04-23T23:41:05.527-01:00</osci:Plain>
          </osci:Creation>
          <osci:Forwarding>
            <osci:Plain>2002-04-24T12:15:01.516-01:00</osci:Plain>
          </osci:Forwarding>
          <osci:Subject>Betreff der Zustellung</osci:Subject>
        </osci:ProcessCard>
        <osci:InspectionReport />
      </osci:ProcessCardBundle>
    </osci:acceptDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:ContentPackage>
      <osci:ContentContainer>
        <osci:Content />
      </osci:ContentContainer>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

An acceptance request (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for acceptance requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="AcceptDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - acceptance request SOAP-Envelope
      $RCSfile: soapAcceptDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>
```

```
  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:acceptDelivery" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>
```

```
</xsd:schema>
```

- Schema for acceptance requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - acceptance request
      $RCSfile: AcceptDelivery.xsd,v $, $Revision: 1.3 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string"
           minOccurs="0" maxOccurs="0"/>
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="prohibited" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="prohibited" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="ProcessCardBundleType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardBundleTemplate">
        <xsd:sequence>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="ProcessCard" type="osci:ProcessCardType" />
          <xsd:element name="InspectionReport"
           type="osci:InspectionReportType" />
        </xsd:sequence>
```

```
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ProcessCardType">
  <xsd:complexContent>
    <xsd:restriction base="osci:ProcessCardTemplate">
      <xsd:sequence>
        <xsd:element name="Creation" type="osci:TimestampType" />
        <xsd:element name="Forwarding"
         type="osci:TimestampType" minOccurs="1" />
        <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="acceptDeliveryType">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultHeaderBlockTemplate">
      <xsd:sequence>
        <xsd:element name="ProcessCardBundle"
         type="osci:ProcessCardBundleType" />
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="IntermediaryCertificatesType">
  <xsd:complexContent>
    <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
      <xsd:sequence>
        <xsd:element name="CipherCertificateIntermediary"
         type="osci:CertificateType" minOccurs="0" />
        <xsd:element name="SignatureCertificateIntermediary"
         type="osci:CertificateType" minOccurs="0" />
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="NonIntermediaryCertificatesType">
  <xsd:complexContent>
    <xsd:extension base="osci:NonIntermediaryCertificatesTemplate" />
  </xsd:complexContent>
</xsd:complexType>

<!-- ### global types ### -->

<!-- ### global elements SOAP-Header ### -->
```

```
  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="acceptDelivery" type="osci:acceptDeliveryType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)
- Adaption of XML Encryption (cf. Section 6.5)

6.6.16 Acceptance Response

An acceptance response is realised as SOAP envelope.

The following example explains the structure of an acceptance response:

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

<!-- XML-Beispiel OSCI 1.2 Annahmeantwort -->
<!-- $RCSfile: ResponseToAcceptDelivery.xml,v $, $Revision: 1.4 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToAcceptDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="0">
      <osci:Response>xaLW8jIJcCAwEA</osci:Response>
    </osci:ControlBlock>
    <osci:responseToAcceptDelivery Id="XREF-0005" soap:mustUnderstand="1"
```

```
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0800</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog beendet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
    </osci:responseToAcceptDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100" />
</soap:Envelope>
```

An acceptance response conforms to the following schemas.

- Adaption of a SOAP envelope for acceptance responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToAcceptDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - acceptance response SOAP-Envelope
      $RCSfile: soapResponseToAcceptDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
```

```
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:responseToAcceptDelivery" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for acceptance responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - acceptance response
      $RCSfile: ResponseToAcceptDelivery.xsd,v $, $Revision: 1.2 $
    </xsd:documentation>
  </xsd:annotation>
```

```
  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" fixed="0" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToAcceptDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:NonIntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateAddressee"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="responseToAcceptDelivery"
   type="osci:responseToAcceptDeliveryType" />
```

```
  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.17 Mediate Delivery Request

A mediate delivery request is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of a mediate delivery request:

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

<!-- XML-Beispiel OSCI 1.2 Abwicklungsauftrag  -->
<!-- $RCSfile: MediateDelivery.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapMediateDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     ConversationId="87634586123" SequenceNumber="5"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:Response>AAQCAEMmCZtuMFQxDSa</osci:Response>
      <osci:Challenge>345nchfcfoqc5dfg</osci:Challenge>
    </osci:ControlBlock>
    <osci:ClientSignature Id="XREF-0001" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <ds:Signature>
        <ds:SignedInfo>
          <ds:CanonicalizationMethod
           Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
          <ds:SignatureMethod
           Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
          <ds:Reference URI="#XREF-0000">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
```

```
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0000/Base64codiert</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#XREF-0002">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0002/Base64codiert</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#XREF-0005">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0005/Base64codiert</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#XREF-0020">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0020/Base64codiert</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#XREF-0100">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0100/Base64codiert</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>Signatur/Base64codiert</ds:SignatureValue>
        <ds:KeyInfo>
          <ds:RetrievalMethod URI="#XREF-9999" />
        </ds:KeyInfo>
```

```
      </ds:Signature>
    </osci:ClientSignature>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
    <osci:mediateDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:ContentReceiver
       URI="http://www.osci.de/2002/04/osci#deployer" />
    </osci:mediateDelivery>
    <osci:NonIntermediaryCertificates Id="XREF-0020"
soap:mustUnderstand="1"
     soap:actor="http://www.w3.org/2001/12/soap-envelope/actor/none">
      <osci:SignatureCertificateOriginator Id="XREF-9999">
        <ds:X509Data>
          <ds:X509Certificate>
            Signierzertifikat/Absender/Base64codiert
          </ds:X509Certificate>
        </ds:X509Data>
      </osci:SignatureCertificateOriginator>
    </osci:NonIntermediaryCertificates>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:ContentPackage>
      <osci:ContentContainer>
        <osci:Content />
      </osci:ContentContainer>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

A mediate delivery request (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for mediate delivery requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="MediateDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
```

```
      OSCI 1.2 - mediate delivery request SOAP-Envelope
      $RCSfile: soapMediateDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:QualityOfTimestamp"
             minOccurs="0" maxOccurs="2" />
            <xsd:element ref="osci:mediateDelivery" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
```

```
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for mediate delivery requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - mediate delivery request
      $RCSfile: MediateDelivery.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="mediateDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="ContentReceiver">
            <xsd:complexType>
```

```
              <xsd:attribute name="URI" type="xsd:anyURI" />
            </xsd:complexType>
          </xsd:element>
          <xsd:sequence minOccurs="0">
            <xsd:element name="MessageId" type="osci:MessageIdType" />
            <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
          </xsd:sequence>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:extension base="osci:NonIntermediaryCertificatesTemplate" />
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="QualityOfTimestamp"
   type="osci:QualityOfTimestampType" />

  <xsd:element name="mediateDelivery" type="osci:mediateDeliveryType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)
- Adaption of XML Encryption (cf. Section 6.5)

6.6.18 Mediate Delivery Response

A mediate delivery response is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of a mediate delivery response:

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

<!-- XML-Beispiel OSCI 1.2 Abwicklungsantwort -->
<!-- $RCSfile: ResponseToMediateDelivery.xml,v $, $Revision: 1.6 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToMediateDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     ConversationId="87634586123" SequenceNumber="6">
      <osci:Response>345nchfcfoqc5dfg</osci:Response>
      <osci:Challenge>fie9uf6dGGFRGm3</osci:Challenge>
    </osci:ControlBlock>
    <osci:SupplierSignature Id="XREF-0001" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <ds:Signature>
        <ds:SignedInfo>
          <ds:CanonicalizationMethod
           Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315" />
          <ds:SignatureMethod
           Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
          <ds:Reference URI="#XREF-0000">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0000/Base64codiert</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#XREF-0005">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0005/Base64codiert</ds:DigestValue>
```

```
          </ds:Reference>
          <ds:Reference URI="#XREF-0010">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0010/Base64codiert</ds:DigestValue>
          </ds:Reference>
          <ds:Reference URI="#XREF-0100">
            <ds:Transforms>
              <ds:Transform
               Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
/>
            </ds:Transforms>
            <ds:DigestMethod
             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
            <ds:DigestValue>Hash/XREF0100/Base64codiert</ds:DigestValue>
          </ds:Reference>
        </ds:SignedInfo>
        <ds:SignatureValue>Signatur/Base64codiert</ds:SignatureValue>
        <ds:KeyInfo>
          <ds:RetrievalMethod URI="#XREF-9999" />
        </ds:KeyInfo>
      </ds:Signature>
    </osci:SupplierSignature>
    <osci:responseToMediateDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0801</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog weiterhin
geöffnet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
    </osci:responseToMediateDelivery>
    <osci:IntermediaryCertificates Id="XREF-0010" soap:mustUnderstand="1"
     soap:actor="http://www.w3.org/2001/12/soap-envelope/actor/none">
      <osci:SignatureCertificateIntermediary Id="XREF-9999">
        <ds:X509Data>
          <ds:X509Certificate>
            Signierzertifikat/Intermediaer/Base64codiert
          </ds:X509Certificate>
        </ds:X509Data>
      </osci:SignatureCertificateIntermediary>
    </osci:IntermediaryCertificates>
  </soap:Header>
  <soap:Body Id="XREF-0100">
```

```
    <osci:ContentPackage>
      <osci:ContentContainer>
        <osci:Content />
      </osci:ContentContainer>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

The element `osci:RequestProcessCardBundle` contains the current process card for the delivery that the user has sent to the intermediary within the *mediate delivery request*.

It only exists if all of the following conditions are met:

- The user has sent a MessageId within the *mediate delivery request.*
- The intermediary was able to create a process card for the delivery within the *mediate delivery request* of the user.

The element `osci:ReplyProcessCardBundle` contains the current process card for the delivery that the service provider has sent to the intermediary within the *processing response*.

It only exists, if at least the following conditions are met:

- Within the processing response, the service provider has sent a delivery with MessageId.
- The intermediary was able to create a process card for the delivery of the service provider.

A mediate delivery response (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for mediate delivery responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToMediateDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - mediate delivery response SOAP-Envelope
      $RCSfile: soapResponseToMediateDelivery.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">
```

```
    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:responseToMediateDelivery" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" minOccurs="0" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for mediate delivery responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - mediate delivery response
      $RCSfile: ResponseToMediateDelivery.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="required" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="required" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="RequestProcessCardBundleType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardBundleTemplate">
        <xsd:sequence>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="ProcessCard"
           type="osci:RequestProcessCardType" />
          <xsd:element name="InspectionReport"
           type="osci:InspectionReportType" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="RequestProcessCardType">
```

```
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardTemplate">
        <xsd:sequence>
          <xsd:element name="Creation" type="osci:TimestampType" />
          <xsd:element name="Forwarding"
           type="osci:TimestampType" minOccurs="1" />
          <xsd:element name="Reception" type="osci:TimestampType"
           minOccurs="1" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseProcessCardBundleType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardBundleTemplate">
        <xsd:sequence>
          <xsd:element name="MessageId" type="osci:MessageIdType" />
          <xsd:element name="ProcessCard"
           type="osci:responseProcessCardType" />
          <xsd:element name="InspectionReport"
           type="osci:InspectionReportType" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseProcessCardType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ProcessCardTemplate">
        <xsd:sequence>
          <xsd:element name="Creation" type="osci:TimestampType" />
          <xsd:element name="Forwarding" type="osci:TimestampType"
           minOccurs="1" />
          <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToMediateDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
          <xsd:sequence minOccurs="0">
            <xsd:element name="RequestProcessCardBundle"
             type="osci:RequestProcessCardBundleType" />
```

```
            <xsd:element name="responseProcessCardBundle"
             type="osci:responseProcessCardBundleType" minOccurs="0" />
          </xsd:sequence>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="IntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
        <xsd:sequence>
          <xsd:element name="SignatureCertificateIntermediary"
           type="osci:CertificateType" minOccurs="0" />
        </xsd:sequence>
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:extension base="osci:NonIntermediaryCertificatesTemplate" />
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="responseToMediateDelivery"
   type="osci:responseToMediateDeliveryType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)

6.6.19 Processing Request

A processing request is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of a processing request:

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

<!-- XML-Beispiel OSCI 1.2 Bearbeitungsauftrag  -->
<!-- $RCSfile: ProcessDelivery.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapProcessDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:Challenge>xaLW8jIJcCAwEA</osci:Challenge>
    </osci:ControlBlock>
    <osci:DesiredLanguages Id="XREF-0002" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next"
     LanguagesList="de en-US" />
    <osci:processDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
    </osci:processDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:ContentPackage>
      <osci:ContentContainer>
        <osci:Content />
      </osci:ContentContainer>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

A processing request (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for processing requests:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
```

```
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ProcessDelivery.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - processing request SOAP-Envelope
      $RCSfile: soapProcessDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:ClientSignature" minOccurs="0" />
            <xsd:element ref="osci:DesiredLanguages" />
            <xsd:element ref="osci:processDelivery" />
            <xsd:element ref="osci:IntermediaryCertificates"
             minOccurs="0" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>
```

```
    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for processing requests:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - processing request
      $RCSfile: ProcessDelivery.xsd,v $, $Revision: 1.4 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string"
           minOccurs="0" maxOccurs="0"/>
          <xsd:element name="Challenge" type="xsd:string" minOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="prohibited" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="prohibited" />
```

```
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ProcessCardBundleType">
  <xsd:complexContent>
    <xsd:restriction base="osci:ProcessCardBundleTemplate">
      <xsd:sequence>
        <xsd:element name="MessageId" type="osci:MessageIdType" />
        <xsd:element name="ProcessCard" type="osci:ProcessCardType" />
        <xsd:element name="InspectionReport"
         type="osci:InspectionReportType" />
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="ProcessCardType">
  <xsd:complexContent>
    <xsd:restriction base="osci:ProcessCardTemplate">
      <xsd:sequence>
        <xsd:element name="Creation" type="osci:TimestampType" />
        <xsd:element name="Forwarding"
         type="osci:TimestampType" minOccurs="1" />
        <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="processDeliveryType">
  <xsd:complexContent>
    <xsd:extension base="osci:DefaultHeaderBlockTemplate">
      <xsd:sequence>
        <xsd:element name="ProcessCardBundle"
         type="osci:ProcessCardBundleType" minOccurs="0" />
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="IntermediaryCertificatesType">
  <xsd:complexContent>
    <xsd:restriction base="osci:IntermediaryCertificatesTemplate">
      <xsd:sequence>
        <xsd:element name="CipherCertificateIntermediary"
         type="osci:CertificateType" minOccurs="0" />
        <xsd:element name="SignatureCertificateIntermediary"
         type="osci:CertificateType" minOccurs="0" />
      </xsd:sequence>
```

```
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:extension base="osci:NonIntermediaryCertificatesTemplate" />
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="ClientSignature" type="osci:SignatureBlockType" />

  <xsd:element name="DesiredLanguages" type="osci:DesiredLanguagesType" />

  <xsd:element name="processDelivery" type="osci:processDeliveryType" />

  <xsd:element name="IntermediaryCertificates"
   type="osci:IntermediaryCertificatesType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)
- Adaption of XML Encryption (cf. Section 6.5)

6.6.20 Processing Response

A processing response is realised either as SOAP envelope or (if it contains attachments) as SOAP message package according to [SOAP A].

The following example explains the structure of a processing response:

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

<!-- XML-Beispiel OSCI 1.2 Abwicklungsauftrag  -->
<!-- $RCSfile: ResponseToProcessDelivery.xml,v $, $Revision: 1.7 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
```

```
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapResponseToProcessDelivery.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Header>
    <osci:ControlBlock Id="XREF-0000" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:Response>xaLW8jIJcCAwEA</osci:Response>
    </osci:ControlBlock>
    <osci:responseToProcessDelivery Id="XREF-0005" soap:mustUnderstand="1"
     soap:actor="http://schemas.xmlsoap.org/soap/actor/next">
      <osci:acknowledgement>
        <osci:Entry xml:lang="de">
          <osci:Code>0800</osci:Code>
          <osci:Text>Auftrag ausgeführt, Dialog beendet</osci:Text>
        </osci:Entry>
      </osci:acknowledgement>
    </osci:responseToProcessDelivery>
  </soap:Header>
  <soap:Body Id="XREF-0100">
    <osci:ContentPackage>
      <osci:ContentContainer>
        <osci:Content />
      </osci:ContentContainer>
    </osci:ContentPackage>
  </soap:Body>
</soap:Envelope>
```

A service provider has to include the element `osci:MessageId` in his response if and only if in the corresponding processing request he has received a delivery with process card.

A processing response (or more precisely its root-body-part) conforms to the following schemas.

- Adaption of a SOAP envelope for processing responses:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.osci.de/2002/04/osci"
   schemaLocation="ResponseToProcessDelivery.xsd" />
```

```
  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - processing response SOAP-Envelope
      $RCSfile: soapResponseToProcessDelivery.xsd,v $, $Revision: 1.5 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Header" minOccurs="1" />
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Header" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Header">
          <xsd:sequence>
            <xsd:element ref="osci:ControlBlock" />
            <xsd:element ref="osci:SupplierSignature" minOccurs="0" />
            <xsd:element ref="osci:QualityOfTimestamp"
             minOccurs="0" maxOccurs="2" />
            <xsd:element ref="osci:responseToProcessDelivery" />
            <xsd:element ref="osci:NonIntermediaryCertificates"
             minOccurs="0" />
            <xsd:any namespace="##other" minOccurs="0"
maxOccurs="unbounded"
             processContents="lax" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:sequence>
            <xsd:element ref="osci:ContentPackage" minOccurs="0" />
          </xsd:sequence>
          <xsd:attribute name="Id" type="xsd:ID" use="required" />
```

```
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Schema for processing responses:

```
<xsd:schema targetNamespace="http://www.osci.de/2002/04/osci"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:include schemaLocation="order.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - processing response
      $RCSfile: ResponseToProcessDelivery.xsd,v $, $Revision: 1.3 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### derived types ### -->

  <xsd:complexType name="ControlBlockType">
    <xsd:complexContent>
      <xsd:restriction base="osci:ControlBlockTemplate">
        <xsd:sequence>
          <xsd:element name="Response" type="xsd:string" minOccurs="1" />
          <xsd:element name="Challenge" type="xsd:string"
           minOccurs="0" maxOccurs="0" />
        </xsd:sequence>
        <xsd:attribute name="ConversationId" type="osci:Number"
         use="prohibited" />
        <xsd:attribute name="SequenceNumber" type="osci:Number"
         use="prohibited" />
      </xsd:restriction>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="responseToProcessDeliveryType">
    <xsd:complexContent>
      <xsd:extension base="osci:DefaultHeaderBlockTemplate">
        <xsd:sequence>
```

```
          <xsd:element name="acknowledgement"
type="osci:acknowledgementType" />
          <xsd:sequence minOccurs="0">
            <xsd:element name="MessageId" type="osci:MessageIdType" />
            <xsd:element name="Subject" type="xsd:string" minOccurs="0" />
          </xsd:sequence>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="NonIntermediaryCertificatesType">
    <xsd:complexContent>
      <xsd:extension base="osci:NonIntermediaryCertificatesTemplate" />
    </xsd:complexContent>
  </xsd:complexType>

  <!-- ### global types ### -->

  <!-- ### global elements SOAP-Header ### -->

  <xsd:element name="ControlBlock" type="osci:ControlBlockType" />

  <xsd:element name="SupplierSignature" type="osci:SignatureBlockType" />

  <xsd:element name="QualityOfTimestamp"
   type="osci:QualityOfTimestampType" />

  <xsd:element name="responseToProcessDelivery"
   type="osci:responseToProcessDeliveryType" />

  <xsd:element name="NonIntermediaryCertificates"
   type="osci:NonIntermediaryCertificatesType" />

  <!-- ### global elements SOAP-Body ### -->

  <xsd:element name="ContentPackage" type="osci:ContentPackageType" />

</xsd:schema>
```

- Adaption of XML Signature (cf. Section 6.4)
- Adaption of XML Encryption (cf. Section 6.5)

6.7 OSCI Messages

OSCI Transport distinguishes between three types of OSCI messages:

- *Encrypted request data* (cf. Section 6.7.1)
- *Not encrypted request data* (cf. Section 6.7.2)
- *Error message* (cf. Section 6.7.3)

6.7.1 Encrypted Request Data

An OSCI message of type *encrypted request data* is realised as SOAP message package according to [SOAP A].

The following example explains the structure of an OSCI message of this type:

```
MIME-Version: 1.0
Content-Type: Multipart/Related; boundary=Trenner; type=text/xml

--Trenner
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <Content-ID@message>

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

<!-- XML-Beispiel OSCI 1.2 Verschluesselte Nachricht -->
<!-- $RCSfile: MessageEncrypted.xml,v $, $Revision: 1.3 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
 xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapMessageEncrypted.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Body>
    <xenc:EncryptedData MimeType="Multipart/Related">
      <xenc:EncryptionMethod
       Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc" />
      <ds:KeyInfo>
        <xenc:EncryptedKey>
          <xenc:EncryptionMethod
           Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5">
            <xenc:KeySize>1024</xenc:KeySize>
          </xenc:EncryptionMethod>
          <ds:KeyInfo>
            <ds:X509Data>
              <ds:X509Certificate>
                Chiffrierzertifikat/Nachrichtenempfaenger/Base64codiert
              </ds:X509Certificate>
            </ds:X509Data>
          </ds:KeyInfo>
          <xenc:CipherData>
            <xenc:CipherValue>
              Verschluesselter/Containerschluessel/Base64codiert
            </xenc:CipherValue>
          </xenc:CipherData>
        </xenc:EncryptedKey>
```

```
      </ds:KeyInfo>
      <xenc:CipherData>
        <xenc:CipherReference URI="cid:Content-ID@attachment" />
      </xenc:CipherData>
    </xenc:EncryptedData>
  </soap:Body>
</soap:Envelope>
--Trenner
Content-Type: text/base64
Content-Transfer-Encoding: 7bit
Content-ID: <Content-ID@attachment>

Encrypted SOAP message package with request data
--Separator
```

An OSCI message of the type *encrypted request data* consists of exactly two body-parts: the root-body-part and a body-part with content-type `text/base64` containing the base64-encoded encrypted request data.

The attribute `URI` of `xenc:CipherData` refers to the content-ID of the body-part with the encrypted request data.

The root-body-part of an OSCI-message of the type *encrypted request data* conforms to the following schemas.

- Adaption of a SOAP envelope for the message type *encrypted request data*:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.w3.org/2001/04/xmlenc#"
   schemaLocation="http://www.w3.org/TR/xmlenc-core/xenc-schema.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - encrypted message SOAP-Envelope
      $RCSfile: soapMessageEncrypted.xsd,v $, $Revision: 1.2 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
```

```
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
          <xsd:choice>
            <xsd:element ref="xenc:EncryptedData" />
          </xsd:choice>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

- Adaption of XML Encryption (cf. Section 6.5)

6.7.2 Not Encrypted Request Data

An OSCI message of the type *not encrypted request data* consists of exactly one request or exactly one response (cf. Section 6.6).

6.7.3 Error Message

An OSCI message of the type *error message* is realised as SOAP envelope.

The following example explains the structure of an OSCI message of this type:

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

<!-- XML-Beispiel OSCI 1.2 Fehlernachricht -->
<!-- $RCSfile: MessageFault.xml,v $, $Revision: 1.3 $ -->

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:osci="http://www.osci.de/2002/04/osci"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                     soapMessageFault.xsd
                     http://www.w3.org/2000/09/xmldsig# oscisig.xsd
                     http://www.w3.org/2001/04/xmlenc# oscienc.xsd">
  <soap:Body>
    <soap:Fault>
      <soap:faultcode>soap:Client</soap:faultcode>
```

```
      <soap:faultstring>Auftragsdaten konnten nicht entschlüsselt
werden</soap:faultstring>
      <soap:detail>
        <osci:Code>9201</osci:Code>
      </soap:detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

An OSCI message of the type *error message* conforms to the following schema.

- Adaption of a SOAP envelope for the message type *error message*:

```
<xsd:schema targetNamespace="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
            xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:osci="http://www.osci.de/2002/04/osci"
            attributeFormDefault="unqualified"
            elementFormDefault="qualified">

  <xsd:import namespace="http://www.w3.org/2001/04/xmlenc#"
   schemaLocation="http://www.w3.org/TR/xmlenc-core/xenc-schema.xsd" />

  <xsd:annotation>
    <xsd:documentation xml:lang="de">
      OSCI 1.2 - error message SOAP-Envelope
      $RCSfile: soapMessageFault.xsd,v $, $Revision: 1.2 $
    </xsd:documentation>
  </xsd:annotation>

  <!-- ### restrictions ### -->

  <xsd:redefine schemaLocation="http://schemas.xmlsoap.org/soap/envelope/">

    <!-- ### Envelope, Header und Body ### -->

    <xsd:complexType name="Envelope" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Envelope">
          <xsd:sequence>
            <xsd:element ref="soap:Body" minOccurs="1" />
          </xsd:sequence>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

    <xsd:complexType name="Body" >
      <xsd:complexContent>
        <xsd:restriction base="soap:Body">
```

```
          <xsd:choice>
            <xsd:element name="Fault" type="soap:Fault" />
          </xsd:choice>
        </xsd:restriction>
      </xsd:complexContent>
    </xsd:complexType>

  </xsd:redefine>

</xsd:schema>
```

7. Conformance Catalogue

In order to conform to this specification, software systems for users, service providers and intermediaries have to observe all the normative parts of this specification that are relevant to them.

Particularly they have to

- use the provided namespaces
- assemble XML documents that conform to the schemas specified in this specification
- observe the reaction rules in Chapter 5 and use the acknowledgement codes specified there

Software systems for intermediaries have to support all the request types in the specified version defined in this specification. Software systems for users and service providers have to support only the request types they need for their specific purpose of application.

8. Reference List

This list of references is divided into a section with references that with regard to this specification have a normative character and another section with references to further reading, included for information purposes.

8.1 Normative References

[ALG 2001] Bundesamt für Sicherheit in der Informationstechnik: Geeignete Kryptoalgorithmen – In Erfüllung der Anforderungen nach §17 (1) SigG vom 16. Mai 2001 in Verbindung mit §17 (2) SigV vom 22. Oktober 1997. Bonn, 5.7.01. Published in: Bundesanzeiger (periodical publication of the German federal authorities) No. 158, 24 August 2001, page 18562. Online available under http://www.bsi.bund.de/esig/basics/techbas/krypto/39.pdf

[EBXML] OASIS ebXML Messaging Services Technical Committee: Message Service Specification. Version 2.0. 1 April 2002

[ISIS-MTT] ISIS-MTT Specification – Common ISIS-MTT Specification for PKI Applications. Version 1.0.1, 15 November 2001. Online available under http://www.t7-isis.de/ISIS-MTT/ISIS-MTT-CoreDocuments-Version1_0_1.pdf

[OSCI-ANF] datenschutz nord GmbH: OSCI-Transport 1.2: Entwurfsprinzipien, Sicherheitsziele und -mechanismen (OSCI Transport 1.2: Design Principles, Security Objectives and Processes). Bremerhaven/Germany, April 2002

[PKCS 1] B. Kaliski, J. Staddon: PKCS #1: RSA Cryptography Specifications – Version 2.0. RFC 2437, October 1998. Online available under http://www.ietf.org/rfc/rfc2437.txt

[PKCS 5] B. Kaliski: PKCS #5: Password-Based Cryptography Specification – Version 2.0. RFC 2898, September 2000. Online available under http://www.ietf.org/rfc/rfc2898.txt

[RFC 2045] N. Freed, N. Borenstein: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. RFC 2045, November 1996. Online available under http://www.ietf.org/rfc/rfc2045.txt

[RFC 2046] N. Freed, N. Borenstein: Part Two: Media Types. RFC 2046, November 1996. Online available under http://www.ietf.org/rfc/rfc2046.txt

[RFC 2111] E. Levinson: Content-ID and Message-ID Uniform Resource Locators. RFC 2111, March 1997. Online available under http://www.ietf.org/rfc/rfc2111.txt

[RFC 2387] E. Levinson: The MIME Multipart/Related Content-type. The Internet Society, RFC 2387, August 1998. Online available under http://www.ietf.org/rfc/rfc2387.txt

[SOAP 1] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. Nielsen, S. Thatte: Simple Object Access Protocol (SOAP) 1.1. W3C Note 08 May 2000. Online available under http://www.w3.org/TR/2000/NOTE-SOAP-20000508. Work in progress! Authoritative for this specification is the version specified, which can differ from the current version (online available under http://www.w3.org/TR/SOAP)

[SOAP A] John J. Barton, Satish Thatte, Henrik Frystyk Nielsen: SOAP Messages with Attachments. W3C Note 11 December 2000. Online available under http://www.w3.org/TR/2001/WD-soap12-part1-20011217/. Work in progress! Authoritative for this specification is the specified version, which may differ from the current version (online available under http://www.w3.org/TR/soap12-part1/)

[XAdES] XML Advanced Electronic Signatures (XAdES). ETSI TS 101 903 V1.1.1 (2002-02)

[XDSIG] Mark Bartel, John Boyer, Barb Fox, Brian LaMacchia, Ed Simon: XML Signature Syntax and Processing. W3C Recommendation 12 February 2002. Online available under http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/

[XENC] Takeshi Imamura, Blair Dillaway, Ed Simon: XML Encryption Syntax and Processing. W3C Candidate Recommendation 04 March 2002. Online available under http://www.w3.org/TR/2002/CR-xmlenc-core-20020304/. Work in progress! Authoritative for this specification is the specified version, which may differ from the current version (online available under http://www.w3.org/TR/xmlenc-core/)

[XML] Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation 6 October 2000. Online available under http://www.w3.org/TR/2000/REC-xml-20001006

[XNAMESP] Namespaces in XML. W3C Recommendation 14 January 1999. Online available under http://www.w3.org/TR/1999/REC-xml-names-19990114

[XSCHEMA 1] XML Schema Part 1: Structures. W3C Recommendation 2 May 2001. Online available under http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XSCHEMA 2] XML Schema Part 2: Data types. W3C Recommendation 02 May 2001. Online available under http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

[XSLT] XSL Transformations (XSLT) – Version 1.0. W3C Recommendation 16 November 1999. Online available under http://www.w3.org/TR/1999/REC-xslt-19991116.html

8.2 Non-Normative References

[SOAP 12P0] SOAP Version 1.2 Part 0: Primer. W3C Working Draft 17 December 2001. Current version is available online under http://www.w3.org/TR/soap12-part0/

[XSCHEMA 0] XML Schema Part 0: Primer. W3C Recommendation, 2 May 2001. Current version is available online under http://www.w3.org/TR/xmlschema-0/

9. Glossary

Acceptance request: → Request by that an → intermediary transmits a → delivery to a → service provider.

Acceptance response: → Response to an → acceptance request.

Attachment: Part of a → delivery. Consists of any kind of data which usually are *not* formulated in XML. Is filled by one or several → authors and is directed to one or several → readers.

Author: Entity that creates the → content data in → content data containers and → attachments.

Business transaction level: Highest level of communication using OSCI Transport. On the business transaction level two → users communicate; one of them in the role of a → sender, the other one in the role of a → recipient.

Challenge value: 1. A random value that a → client puts in his request to a → supplier. The supplier has to repeat this value as a → response value in his response. 2. Within an → explicit dialog with a → client, a random value that a → supplier puts in his response. The client has to repeat this value in his next request.

Client: Role of a → user or an → intermediary on the → request level. The client is the communication partner who initiates the communication. He sends a → request to the → supplier and receives his → response.

Content data: Data that a → user wants to transmit to another user. The transmission of these data gives cause for the use of OSCI Transport.

Content data container: Part of a → delivery. May contain any information formulated in XML. Is filled by one or several → authors and is directed to one or several → readers. May particularly contain encryption information to further content data containers and/or → attachments within the same delivery. Can be signed by the authors.

ConversationId: Identity of a → dialog.

Delivery: Data package on the → business transaction level that a → sender directs to a → recipient. Can contain any number of → content data containers and/or → attachments. Some or all of the content data containers and/or attachments can be encrypted. In this case the delivery contains information that the recipient needs to decrypt the content data containers and/or attachments. The delivery can contain one or several certificates as well.

Delivery request: → Request by that a → user transmits a → delivery to an → intermediary so that the latter will have it ready for collection by another user.

Delivery response: → Response to a → delivery request.

Dialog: Sequence of request-response pairs. Consists of at least one request-response pair. Within a dialog, before sending his next → request the → client has to wait until he will have received the → response of the → supplier upon the previous request.

Dialog exit request: → Request by that a → client an → explicit dialog with a → supplier.

Dialog exit response: → Response to a → dialog exit request.

Dialog init request: → Request by that a → client opens an → explicit dialog with a → supplier.

Dialog init response: → Response to a → dialog init request.

Dialog, explicit: → Dialog that comes into being by a → dialog init request. Continues until the → supplier will have sent his response upon the → dialog exit request of the → client.

Dialog, implicit: Opposite to → explicit dialog. Comes into being by any → request (that is no → dialog init request). Continues until the → supplier will have sent his → response.

Fetch delivery request: → Request by that a → user collects a → delivery at an → intermediary's that another user has handed in before by means of a → delivery request.

Fetch delivery response: → Response to a → fetch delivery request.

Fetch process card request: → Request by that a → user collects one or several → process cards at an → intermediary's.

Fetch process card response: → Response to a → fetch process card request.

Forwarding delivery request: → Request by that a → user transmits a → delivery to an → intermediary, so that the latter will forward it to a → service provider. In contrast to the → mediate delivery requestthe user expects in the response of the intermediary *no* delivery of the service provider.

Forwarding Delivery Response: → Response to a → forwarding delivery request.

Intermediary: Entity that mediates in the information exchange between → users. The intermediary's role is neutral. Physically, however, it may well be located at one of the two communicating users.

Mediate delivery request: → Request by that a → user transmits a → delivery to an → intermediary, so that the latter will forward it to a → service provider. In contrast to the → forwarding delivery request the user expects a delivery of the service provider in the response of the intermediary.

Mediate delivery response: → Response to a → mediate delivery request.

Message level: Base level of the communication using OSCI Transport. On the message level a → user and an → intermediary communicate; one of them in the role of a → message sender, the other one in the role of a → message recipient. Both communication partners can assume both roles.

Message recipient: Role of a → user or an → intermediary on the → message level. A message recipient receives an → OSCI message from a → message sender.

Message sender: Role of a → user or an → intermediary on the → message level. A message sender sends an → OSCI message to a → message recipient.

MessageId: Identity of a → delivery.

MessageId request: → Request by that a → user orders a MessageId at the intermediary's.

MessageId response: → Response to a → MessageId request.

OSCI message: Data package on the → message level that a → message sender directs to a → message recipient. Contains either exactly one → request or exactly one → response or exactly one error message. The request / the response can be encrypted. In this case the OSCI message contains additional information the message recipient needs to decrypt the request / response.

Process card: In a process card the way of a → delivery from a → sender via an → intermediary to a → recipient is recorded. The process card is held by the intermediary.

Processing request: → Request by that an → intermediary asks a → service provider to process a → delivery of a → user and to provide another delivery as a reaction.

Processing response: → Response to a → processing request.

Reader: Entity → content data in → content data containers and → attachments are directed to.

Recipient: Role of a → user on the → business transaction level. A recipient accepts a → delivery of a → sender.

Request: Data package on the → request level that a → client directs to a → supplier. Contains a working instruction for the supplier and further information that the supplier needs to execute the instruction and to send a → response. In particular a request can contain a → delivery. It can be signed by the client.

Request level: Middle level of communication using OSCI Transport. On the request level a → user and an → intermediary communicate; one of them in the role of a → client, the other one in the role of a → supplier. Usually the user is the client and the intermediary is the supplier; there are, however, also scenarios in that the roles are reversed.

Response: Data package on the → request level that a → supplier directs as a reaction upon a → request to a → client. In the response the supplier informs about the result of the processing of the request. In particular a response can contain a → delivery. The response can be signed by the supplier.

Response value: 1. Value that a → supplier puts into his response as a reaction to the → challenge value in the corresponding request of the → client. 2. Within an → explicit dialog with a → supplier, a value that a → client puts in his subsequent request as a reaction to the → challenge value in the response of the supplier.

Sender: Role of a → user on the → business transaction level. A sender directs a → delivery to a → recipient.

SequenceNumber: Number of a request / a response within a → dialog.

Service provider: Special → user who is available (in principle all the time) under a URL.

Supplier: Role of a → user or an → intermediary on the → request level. The supplier is the reacting communication partner. He receives the → request of the → client and sends a → response as a reaction.

Usage data: Data that are exchanged in addition to the → content data and that help to check and control the data flow.

User: Entity that uses OSCI Transport to exchange information with other users. A user can be a person, a group of persons or even a software system. The only prerequisite a user has to fulfill is to possess a valid encryption certificate.

10. Table of Figures