

2007-04-05

SHS Version 1.2.01 DTD Descriptions

Verva - Swedish Administrative Development Agency

Editors:

Kurt Helenelund, Anders Bremsjö, Stephan Urdell, Bo Sehlberg,
Jan Lundh, Christer Marklund

Copyright © 2003,2004,2005, The Swedish Agency for Public
Management, 2007 Swedish Administrative Development Agency. All
Rights Reserved. Verva Swedish Administrative Development Agency
document use and open specification rules apply.

2007-04-05

Contents

1	INTRODUCTION.....	5
1.1	AUDIENCE	5
1.2	REFERENCES.....	5
1.3	DOCUMENT HISTORY	5
1.4	DOCUMENT CONVENTIONS.....	6
2	DTD OVERVIEW.....	8
3	DTD VERSIONS.....	9
3.1	VERSION NUMBERING	9
3.2	COMPATIBILITY	9
3.3	VERSION HANDLING IN THE XML DOCUMENTS.....	9
4	URN SYNTAX.....	10
5	LABEL	11
5.1	INTRODUCTION.....	11
5.2	DESCRIPTION OF ELEMENTS	11
6	AGREEMENT	21
6.1	INTRODUCTION.....	21
6.2	ENTITIES.....	21
6.3	ELEMENTS.....	22
7	PRODUCT.....	39
7.1	INTRODUCTION.....	39
7.2	ENTITIES.....	39
7.3	ELEMENTS.....	39
8	MANAGEMENT.....	47
8.1	SYNCHRONOUS CALLS.....	47
8.2	ERROR CODES	47
8.3	ELEMENTS.....	47
8.4	EXAMPLES.....	50
9	MESSAGE LISTS.....	51
9.1	ELEMENTS.....	51
9.2	EXAMPLES.....	55

2007-04-05

1 Introduction

This document include a description of SHS version 1.2 DTD's documentation.

1.1 Audience

This document is intended for technical SHS administrators, system integrators and developers.

1.2 References

[Protocols]

SHS Version 1.2.01 - Protocol

1.3 Document history

Version	Date	Change	By	Approved
0.1.0	<date>	Agreement.doc, Label.doc and Product.doc	Kurt Helenelund	
0.2.0	1999-04-06	All three documents are put in one.	Ulf Gundersen	
0.3.0	1999-04-09	Included DTD version handling and acknowledgement and error message discussion	David Kågedal	
0.3.1	1999-04-15	Translated the version chapter to English. Removed things from the DTDs that probably are irrelevant, redundant, or simply unnecessary. Added a few comments regarding issues needing clarifications.	David Kågedal	
0.4.0	1999-04-27	Added management.dtd. Decided on DTD version in mechanism Refined confirmation/error handling description. Added intro to Agreement chapter. Minor simplifications to agreement.dtd Moved all BNF grammars to own section in the label chapter. Removed internal-id from the address URN Some major changes to the label DTD.	David Kågedal	
0.4.1	1999-05-17	Translated everything to English	David Kågedal	
0.5.0				
0.6.0	2000-03-14		David Kågedal	
1.0	2000-04-03		David Kågedal	Yvonne Palminger
1.0.1	2000-11-30	Updated according to SHS version 1.1. Main changes are according to section 4.1, 4.2, 5.2, 6.1, 6.2, 6.3 and 6.4. Some other minor adjustments are made as well.	Stephan Urdell	

2007-04-05

1.0.2	2001-03-05	Updated with version 1.1 proposal 6.6, 7.1 and 7.2. Also updated according to SHS ÅR meeting 2000-12-11 and comments from Bo Sehlberg. All instances of attribute <i>CommonName</i> are defined as optional (IMPLIED). Containing change marks for changes compared to document version 1.0.	Stephan Urdell	
1.0.3	2001-06-13	Updated according to comments from Kurt Helenelund and Bo Sehlberg. Setting the SHS transport to always use MIME/SSL and the same MIME version for all parts of a SHS document. Changing "urn:shs:" to "urn:X-shs:". Adding support for variable number of data parts for the same product type. Still containing change marks for changes compared to document version 1.0.	Stephan Urdell	
1.1	2001-09-16	Final 1.1 version minor corrections	Kurt Helenelund	Christer Marklund
1.1.1	2003-03-28	First draft of 1.2 documentation <ul style="list-style-type: none"> Moved acknowledgement and error messages to architecture/protocol Minor changes in existing text 	Anders Lindgren	
1.1.2	2003-05-27	Second draft of 1.2 documentation <ul style="list-style-type: none"> Overview/introduction sections where these were missing Overall change from "SHS document" to "SHS message" 	Anders Lindgren	
1.2	2003-10-09	Final version 1.2	Anders Lindgren	Jan Lundh
1.2.01-A		First draft version (A) with changes in 1.2 specification (1.2.01) <ul style="list-style-type: none"> Clarifications Content ID in message lists Number of data elements in content 	Anders Lindgren	
1.2.01	2004-06-03	Final version 1.2.01	Anders Lindgren	
1.2.01	2007-04-05	Published as Verva document		Christer Marklund Jan Lundh

1.4 Document Conventions

In this document the following conventions are used:

Times New
Roman

Normal text

[REFERENCE]

References are normal text (UPPER CASE) enclosed in square brackets.

Courier 10pt

Technical details

NoteSide

Flag sample text

2007-04-05

URN syntax descriptions are based on an augmented form of Backus Naur Format (BNF) as used in RFC2141.

2007-04-05

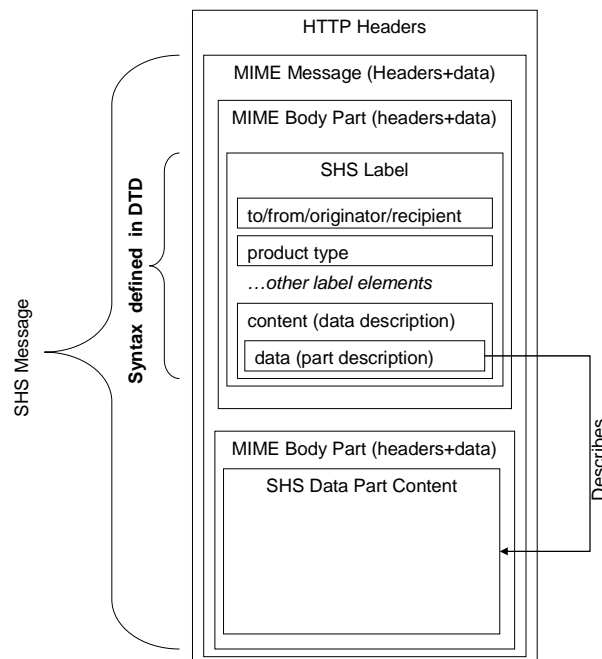
2 DTD Overview

The role of the DTD is to provide a formal description of the content in a given message. The formality enables programs to parse and interpret the description automatically and is therefore helpful when complex data are exchanged between systems. The definition may be stored locally or remote in the network or even be co-located with the information.

Within SHS the DTDs are used for three major purposes:

- Provide a standardized way of handling header information that are used by SHS for transmission and routing decisions (from/to, asynch/synch etc.)
- Provide a standardized way to describe configuration data such as agreements.
- Describe the payload transported by the SHS (product-types etc)

The following picture gives a conceptual view of a simple SHS transmission.



2007-04-05

3 DTD Versions

3.1 Version Numbering

The DTDs used by SHS are versioned by a two-part version number **x.y**.

These two parts corresponds to and is equal to the version and revision part of a complete SHS version specification. The first part (**x**) is the version specification and the second part (**y**) is the revision specification.

3.2 Compatibility

When new DTD versions are introduced, there will be a period where different SHS will handle different DTD versions. A requirement is that a SHS must handle DTD-versions with different revision numbers (**y**) for the same version number (**x**). In this way an SHS node, which has not yet been upgraded, will still be able to send messages to an upgraded SHS.

When the version number (**x**) changes no compatibility requirement is specified. Such version incompatibility issues may instead be handled by some kind of version gateway or converter.

3.3 Version Handling in the XML Documents

Every DTD includes in its document (root) element an attribute named `version`. This attribute denotes which DTD version to which an XML document conforms. However, to be able to determine which DTD version a XML document conforms to, the XML document has to be parsed to find the `version` attribute. This cannot be safely done without knowing in advance which DTD is used. To avoid this chicken-and-egg problem, the DTD is named with a name that includes the version number. In this way the first version of the label DTD will be named `shs-label-1.0.dtd`. This name is then used in the `<DOCTYPE>` declaration in the XML document.

The `<DOCTYPE>` declaration starts with the specification of which element that is the root element in the xml document. After the `SYSTEM` keyword in the `<DOCTYPE>` specification is the filename of the DTD-file specified. This is the physical DTD-file that is used when creating the content corresponding to this `<DOCTYPE>` specification. This filename also includes the version of the DTD.

This is an example xml header specification for the SHS label:

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!DOCTYPE shs.label SYSTEM "shs-label-1.2.dtd">
```

This is an example xml header specification for the SHS message listing:

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!DOCTYPE shs.message-list SYSTEM "shs-message-list-1.2.dtd">
```

2007-04-05

4 URN syntax

A Universal Resource Name, or URN, is a string that identifies a resource in a fully qualified way. The basic syntax for URNs is defined by the IETF in RFC 2141.

URN syntax is used in SHS for naming addresses, actors and product types. For all these purposes, the "shs" namespace is used, and this chapter describes the syntax for the URNs under this namespace. The syntax is defined using a grammar in a simple BNF form similar to that used in RFC 2141.

```

<shs-urn> ::= <shs-urnid> <shs-nss>
<shs-urnid> ::= "urn:" <shs-nid> ":"
<shs-nid> ::= "X-shs"
<shs-nss> ::= <actor> | <product> | <address> | <enduser>

<shs-actor> ::= <shs-urnid> <actor>
<shs-product> ::= <shs-urnid> <product>
<shs-address> ::= <shs-urnid> <address>
<shs-enduser> ::= <shs-urnid> <enduser>

<actor> ::= <shs-chars>+

<product> ::= <uuid> | <special-product>
<uuid> ::= 8<hex>"-"4<hex>"-"4<hex>"-"4<hex>"-"12<hex>
<special-product> ::= "confirm" | "error " | "agreement"

<address> ::= <shs-chars>+ ("." <internal-id>)*
<internal-id> ::= <shs-chars>+

<enduser> ::= <person>|<organization>|<mail-user>|<fax-user>
<person> ::= "pno:" <number>+ (("+" | "-") <number>+)*
<organization> ::= "orgno:" <number>+ (("+" | "-") <number>+)*
<mail-user> ::= "mailto:" <mail-subaddr> "@" <mail-subaddr>
<mail-subaddr> ::= <shs-chars>+ ("." <shs-chars>+)*
<fax-user> ::= "fax:" <shs-chars>+

<shs-chars> ::= <upper> | <lower> | <number> | <shs-other>
<shs-other> ::= "(" | ")" | "+" | "," | "-" | "_"

<hex> ::= Defined by RFC 2141 (0 - 9, A - F, a - f)
<upper> ::= Defined by RFC 2141 (A - Z)
<lower> ::= Defined by RFC 2141 (a - z)
<number> ::= Defined by RFC 2141 (0 - 9)

```

Note that although URN:s use the Unicode character set, they can only contain unencoded ASCII character, so non-ASCII characters need to be encoded using UTF-8, followed by escaping all octets that are not in <URN chars> using the %xx hexadecimal escape syntax. See RFC 2141 for the definitions of <hex>, <upper>, <lower>, <number> and <URN chars>.

2007-04-05

5 Label

5.1 Introduction

This DTD describes the label that is attached to every SHS message.

One of the basic principles for the label is that no application data is included in the label. The label is just for specifying information that is sent together with the data.

The exception to this basic rule is the element *meta*, which may contain data that is specific to a number of applications. The interpretation of such specific data is agreed upon in advance by the applications.

For all messages that are received by a SHS a standard validation is performed. The label attribute *sequence-type* is checked to have a valid value during this validation. The sequence type attribute value is valid if it is one of the predefined values and the value is approved in combination with the *response required* attribute for the corresponding product type definition.

The following table defines the approved combinations of attribute values:

Sequence type	Response required
event	no
request	yes, no
reply	yes, no
adm	no

5.2 Description of Elements

5.2.1 shs.label

Syntax and usage:

```
<!ELEMENT shs.label ((originator, from?) | from), to?,
end-recipient?, product, meta*, subject?,
datetime, content, history*)>

<!ATTLIST shs.label
version CDATA #FIXED "1.2"
tx.id NMTOKEN #REQUIRED
corr.id NMTOKEN #REQUIRED
shs.agreement NMTOKEN #IMPLIED
transfer-type (asynch|synch) "asynch"
message-type (simple|compound) "simple"
document-type (simple|compound) "simple"
sequence-type (event|request|reply|adm) #REQUIRED
status (test|production) "production" >
```

2007-04-05

Description

The root element *shs.label* contains information of how to manage the content of a SHS message. A label is attached to every SHS message and it is always placed first in the message.

If the message only contains an *originator* element and no *from* element when it reaches the first SHS, this first receiving SHS must add a *from* element before the message is given to a new receiver. A recommended value to supply as *from* element is the actor that controls the SHS. The value may also contain an internal part to make an eventual return address specification easier.

The *to* element may be omitted. Which SHS that should handle the message next in this case is resolved in the first SHS by using local and public agreements. The first SHS may also be the resolved handler and receiver of the message.

Elements:

<code>originator</code>	Specifies the end user that created the message content from the beginning.
<code>from</code>	Specifies the actor that sent the message.
<code>to</code>	Specify the actor that is to receive the message.
<code>end-recipient</code>	Specifies the end user that will be the final recipient of the message.
<code>product</code>	Specifies the product type that the message contains.
<code>meta</code>	Message content annotations.
<code>subject</code>	Short description of message content.
<code>datetime</code>	The time stamp for the label creation according to ISO 8601 extended format 'yyyy-mm-ddThh:mm:ss'. This is the message time stamp.
<code>content</code>	describes the contents
<code>history</code>	describes the history of the message

Attributes:

<code>version</code>	Specifies the version of SHS label dtd that an instance of the <i>shs.label</i> root element conforms to. This attribute is used together with the DOCTYPE specification to decide the version of dtd to use when reading and parsing the <i>shs.label</i> element.
<code>tx.id</code>	Specifies the uuid for a message transaction.

2007-04-05

<code>corr.id</code>	specifies the correlation identity so that the sender application can associate any received response with a query. This is a string specified by the sender.
<code>shs.agreement</code>	specifies which agreement is used for the exchange
<code>transfer-type</code>	specifies whether the connection is synchronous or asynchronous
<code>message-type</code>	specifies whether it is a simple SHS message or a combination of messages
<code>document-type</code>	same information as <code>message-type</code> supplied only for backward compatibility with SHS 1.1 implementations
<code>sequence-type</code>	Specifies what part this message plays in the information exchange. This attribute is orthogonal to the <code>transfer-type</code> attribute, which means among other things that both synchronous and asynchronous queries and responses can occur.
<code>event</code>	The message is a simple transfer, for example a file transfer. No response at application level is allowed.
<code>request</code>	The message is one of a query/reply pair. An application message with the requested content is expected in a message addressed in the opposite direction, and with the <code>sequence-type</code> set to "reply". The reply message is optional if the corresponding product type attribute <code>resp-required</code> is set to "no".
<code>reply</code>	The message is the reply for a query/reply pair.
<code>adm</code>	The message is an administrative message. The SHS special product type used for the message specifies which administrative information that is contained in the message.
<code>status</code>	flag to indicate whether this is a test or a production

5.2.2 originator

Syntax and usage:

```
<!ELEMENT   originator   (#PCDATA) >
<!ATTLIST  originator
           name           CDATA          #IMPLIED
           labeledURI     CDATA          #IMPLIED >
```

Description

Specifies the end user that created the message content from the beginning. The `<shs-enduser>` production in chapter 4 defines the syntax for the element value.

2007-04-05

Attributes:

name	The name of the originator. This name is used to improve readability.
labeledURI	Pointer to further information about the originator.

5.2.3 from

Syntax and usage:

```
<!ELEMENT from (#PCDATA)>
<!ATTLIST from
  commonName CDATA #IMPLIED
  labeledURI CDATA #IMPLIED >
```

Description

This element defines the sender of the message. The element value contains a unique identity for the sender. The <shs-address> production in chapter 4 defines the syntax for the element value.

Attributes:

commonName	the common name of the sender. A name used to improve human readability.
labeledURI	optional pointer to further information about the sender

5.2.4 to

Syntax and usage:

```
<!ELEMENT to (#PCDATA)>
<!ATTLIST to
  commonName CDATA #IMPLIED >
```

Description

This element defines the receiver of the message. The element value contains a unique identity for the receiver. The <shs-address> production in chapter 4 defines the syntax for the element value.

Attributes:

commonName	The common name of the receiver. Note that the sender can specify a name for the receiver.
------------	--

2007-04-05

5.2.5 End-recipient

Syntax and usage:

```
<!ELEMENT end-recipient (#PCDATA) >
<!ATTLIST end-recipient
    name CDATA #IMPLIED
    labeledURI CDATA #IMPLIED >
```

Description

Specifies the end user that will be the final recipient of the message.

Attributes:

name	The name of the end recipient. This name is used to improve readability.
labeledURI	Pointer to further information about the end recipient.

5.2.6 product

Syntax and usage:

```
<!ELEMENT product (#PCDATA) >
<!ATTLIST product
    commonName CDATA #IMPLIED
    labeledURI CDATA #IMPLIED >
```

Description

This element defines the product type of the message. The element value contains a unique identity (URN) for the product type that the label is defined for. The <shs-product> production in chapter 4 defines the syntax for the element value.

Attributes:

commonName	A human-readable short description of the product type. A name used to improve human readability. This attribute is not used identify the product type; the URN is used for that.
labeledURI	Pointer to further information about the product type.

2007-04-05

5.2.7 meta

Syntax and usage:

```
<!ELEMENT meta (#PCDATA) >
<!ATTLIST meta
  name CDATA #REQUIRED >
```

Description

This element makes it possible to create an annotation for the message content. It consists of a name-value pair. This element is intended to be a short non-verbal annotation or description and to be used mostly for automatic computer reading and interpretation.

The element value may contain any form of information that is agreed upon by the different parts that are handling this element. For a specific name the value of this element often will be chosen from a list of enumerated possible values.

This element can be used to implement content dependent decisions. If desirable and needed, it can also be used to create different sort of subgroups within one product type.

Attributes:

name	The name of the content annotation.
------	-------------------------------------

5.2.8 subject

Syntax and usage:

```
<!ELEMENT subject (#PCDATA) >
```

Description

This element contains a description of the message content. This is a verbal description intended for human reading. This information is application specific and has no control function for the SHS.

The element may contain any form of information that is descriptive for the content of the message.

5.2.9 datetime

Syntax and usage:

```
<!ELEMENT datetime (#PCDATA) >
```


2007-04-05

Description

Contains a date and time specification. The value is constructed according to ISO 8601 extended format 'yyyy-mm-ddThh:mm:ss'.

5.2.10 content

```
<!ELEMENT content ((data*|compound), comment?) >
<!ATTLIST content
content.id NMTOKEN #REQUIRED>
```

Elements:

- `data` Specifies that there is a data part in the message. For each data part description in the label the data part type name is used to specify which type of data part that is described. The actual data are stored outside the label as one or several separate MIME parts.
- `compound` Specifies that this is a compound SHS message. Each data part that is found in the message is in turn a complete message with its own label.
- `comment` Optional verbal comment and description for the content.

Attributes:

- `content.id` A unique identity of the content. This unique ID identifies the information in the data, meaning that if the same information is resent in another SHS message, as a new transaction, the same content id may be used. This is a string specified by the sender.

5.2.11 data

Syntax and usage:

```
<!ELEMENT data EMPTY >
<!ATTLIST data
datapartType CDATA #REQUIRED
filename CDATA #IMPLIED
no-of-bytes NMTOKEN #IMPLIED
no-of-records NMTOKEN #IMPLIED >
```

Description

Specifies one data part that is part of current message.

The data part type attribute is copied from the product type definition to the message when the message label is created. This gives a reference between the corresponding data part specification and a specific data part in a message.

2007-04-05

Attributes:

<code>datapartType</code>	Specifies the type of data part for this data part. Should be a copy of the data part type attribute from the corresponding data part description in the product type definition. Is intended for automatic computer reading and interpretation.
<code>filename</code>	The name of the file that is attached. This could be used when the file is saved on disk so that it has the same name as when it was sent. It is up to the person that constructs the label to select a file name. When the data in the message does not originally come from a file, a random name can be chosen to write in the attribute or the name can be omitted.
<code>no-of-bytes</code>	specifies the size of the data
<code>no-of-records</code>	specifies the number of records, if a record structure is used

5.2.12 compound

Syntax and usage:

```
<!ELEMENT compound EMPTY >
<!ATTLIST compound
  no-of-parts NMTOKEN #REQUIRED >
```

Attributes:

<code>no-of-parts</code>	The number of messages that are included in the compound message. The value may be zero or a positive integer.
--------------------------	--

5.2.13 comment

Syntax and usage:

```
<!ELEMENT comment (#PCDATA) >
```

Description

Specifies a comment. The value is a text field intended for comments.

2007-04-05

5.2.14 history

Syntax and usage:

```
<!ELEMENT history (datetime, comment?) >
<!ATTLIST history
  node.id NMTOKEN #REQUIRED
  content.id NMTOKEN #IMPLIED
  tx.id NMTOKEN #IMPLIED
  from NMTOKEN #IMPLIED
  to NMTOKEN #IMPLIED
  local.id NMTOKEN #IMPLIED
  shs.agreement NMTOKEN #IMPLIED >
```

Description

This element is used for tracing the flow and handling of the message. New elements are created at the entry to a new SHS node or when the value of any history attribute has changed. A new history element is always added after all old existing history elements.

When a new element is created at the entry to a new SHS node it contains values for all attributes that have defined values.

When a new element is created because of attribute changes within the same SHS node it only contains values for those attributes that has changed. In this case the history attribute contains the old attribute value.

When a message is redistributed to a number of new receivers, the existing history elements in the current message still will be present in the messages sent to the new receivers.

When a message is sent to a number of new receivers because of subscription, the existing history elements in the current message are not present in the messages sent to the subscribers. In this case the messages sent to the subscribers are treated as new messages with the same content (data parts) as the original message.

Elements:

datetime	a time stamp
comment	optional comments (text)

Attributes:

node.id	SHS node identity
content.id	Identity for the content.
tx.id	uuid for message transactions

2007-04-05

<code>from</code>	The element value from the <from> element in the label.
<code>to</code>	The element value from the <to> element in the label.
<code>local.id</code>	A local identity that has been assigned for the message. This assignment is done by the current SHS.
<code>shs.agreement</code>	unique identity for the agreement to use for the exchange

2007-04-05

6 Agreement

6.1 Introduction

This DTD describes how an agreement XML description is composed. The agreement description is used to describe the agreement between two parties having signed a contract for information interchange. The agreement description is used for several purposes:

- The SHS systems involved use information to determine when and where to send data, to collect billing information, etc.
- The applications sending to and fetching from a SHS may use it.
- Some parts of the agreement are purely informational.

6.2 Entities

6.2.1 HrefAtt

Syntax and usage:

```
<!ENTITY % HrefAtt 'CDATA #IMPLIED'>
```

Description:

This entity is used to keep URI ‘pointers’ to external references.

6.2.2 TimeQuantAttList

Syntax and usage:

```
<!ENTITY % TimeQuantAttList '
    day      NMTOKEN #IMPLIED
    hour     NMTOKEN #IMPLIED
    min      NMTOKEN #IMPLIED'>
```

Description:

This entity is used to specify time periods.

6.2.3 TimeUnit

Syntax and usage:

```
<!ENTITY % TimeUnit '(sec|min|hour|day|week|month|year)'>
```

2007-04-05

Description:

This entity is used to specify periods of time.

6.2.4 SizeUnit

Syntax and usage:

```
<!ENTITY % SizeUnit '(B|KB|MB|GB) '>
```

Description:

This entity is used to specify the unit of data size specifications.

B = Byte

KB = KiloByte (1024 Byte)

MB = MegaByte (1024*1024 Byte)

GB = GigaByte (1024*1024*1024 Byte)

6.2.5 DaySpec

Syntax and usage:

```
<!ENTITY % DaySpec '(Mon|Tue|Wen|Thu|Fri|Sat|Sun|  
Mon-Fri|Sat-Sun|work|holiday|every) '>
```

Description:

This entity is used to specify a day of week, a kind of day of week or *every* if the day of week is irrelevant.

6.3 Elements

The elements specified for the agreement are:

- agreement - root element
 - shs - shs specific parameters
 - principal - information about the principal, who is the 'owner' of the agreement
 - product - information about the product type that the agreement concerns
 - customer - unique identity for the customer that the agreement is set up for
 - direction – direction of information flow

2007-04-05

- billing – if billing information should be collected this element specifies how
 - confirm - Describes when the delivery confirmation is used
 - error - information about how errors should be handled
- general - general information for information exchange between two parts
 - description
 - valid – validation period for agreement
 - schedule – times for transfer
 - QoS - of service requirements at service level

6.3.1 shs.agreement

Syntax and usage:

```
<!ELEMENT shs.agreement (shs, general)>
<!ATTLIST shs.agreement
  version CDATA #FIXED "1.2"
  uuid NMTOKEN #REQUIRED
  transfer-type (asynch|synch|any) #REQUIRED
  contract %HrefAtt;>
```

Description

The root element shs.agreement contains information that describes an agreement concerning information exchange between two parts.

Elements:

shs	Contains the agreement information that is dedicated for shs specific details.
general	Contains the agreement information that handles general information exchange parameters.

Attributes:

version	Specifies the version of SHS agreement dtd that an instance of the shs.agreement root element conforms to. This attribute is used together with the DOCTYPE specification of an agreement XML document to decide the version of dtd to use when reading and parsing the shs.agreement element.
---------	--

2007-04-05

<code>uuid</code>	Unique identity for the agreement.
<code>transfer-type</code>	Specifies the type of transfer that this agreement permits. If any is specified, both asynchronous and synchronous transfer may be used.
<code>contract</code>	Pointer to an external reference for the contract. This is an URL that can point to the contract. It can also point to additional information about the contract. This additional information may or may not be contained in the agreement.

6.3.2 shs

Syntax and usage:

```
<!ELEMENT shs (principal, product+, customer?, direction, confirm?, billing?, error?)>
```

Description

The element `shs` contains information that describes `shs` specific parameters for information exchange between two parts. The `<principal>` part is the one that owns the agreement for the product types that are to be exchanged.

The `customer` element may be missing (unspecified) when the agreement direction is from customer (to principal). In all other cases the `customer` element must be specified. The missing element specifies that any customer may use this agreement.

<code><principal>-- <product>-- ><customer></code>	from principal to customer
<code><principal>< -- <product>--<customer></code>	from customer to principal
<code><principal>< -- <product>-- > <customer></code>	either way
<code><principal> req- <product> -- > <customer></code> <code><principal> < -- <product> -rep <customer></code>	Query/reply initiated by principal
<code><principal> < -- <product> -req <customer></code> <code><principal> rep- <product> -- > <customer></code>	Query/reply initiated by customer

Note that `query/reply` is a product property that is described in the SHS-product DTD.

Elements:

<code>principal</code>	The owner of the agreement.
<code>product</code>	The product types that are to be exchanged.
<code>customer</code>	The customer that leaves or receives the product.

2007-04-05

<code>direction</code>	the direction of the flow for the product
<code>confirm</code>	describes the appearance of the confirmation
<code>billing</code>	invoicing information
<code>error</code>	Describes error handling

6.3.3 principal

Syntax and usage:

```
<!ELEMENT principal (#PCDATA) >
<!ATTLIST principal
  commonName CDATA #IMPLIED
  labeledURI %HrefAtt; >
```

Description

Identity and ‘pointer’ to information about the principal, who is the ‘owner’ of the agreement. The element value contains a unique identity for the principal. The `<shs-actor>` production in chapter 4 defines the syntax for the element value.

Attributes:

<code>commonName</code>	Trivial name for the principal. This is only used to improve readability, and should be a human-readable name, preferably the same as the <i>organizationName</i> in the actor catalogue.
<code>labeledURI</code>	URI to more information about <code><principal></code> . This can be the organization’s WWW homepage or anything else.

6.3.4 product

Syntax and usage:

```
<!ELEMENT product (#PCDATA) >
<!ATTLIST product
  commonName CDATA #IMPLIED
  labeledURI %HrefAtt; >
```

Description

Identity and ‘pointer’ to information about the product type that the agreement concerns. The element value contains a unique identity for the product that the agreement is set up for. The `<shs-product>` production in chapter 4 defines the syntax for the element value.

2007-04-05

This may reference the SHS special product types declared in chapter 4, e.g. **confirm**. Although it is not required to use agreements for these special product types it is still accepted.

Attributes:

<code>commonName</code>	Trivial name for the product type. A name used to improve human readability. The <code>commonName</code> does not uniquely identify the product type. The value is preferably the same as the <i>productName</i> in the product catalogue.
<code>labeledURI</code>	Pointer to more information about the product type

6.3.5 customer

Syntax and usage:

```
<!ELEMENT customer (#PCDATA) >
<!ATTLIST customer
  commonName CDATA #IMPLIED
  labeledURI %HrefAtt; >
```

Description

A unique identity for the customer that the agreement is set up for. The `<shs-actor>` production in chapter 4 defines the syntax for the element value.

Attributes:

<code>commonName</code>	The common name for the customer. This is only used to improve readability, and should be a human-readable name, preferably the same as the <i>organizationName</i> in the actor catalogue.
<code>labeledURI</code>	URI to more information about <code><customer></code> . This can be the organization's WWW homepage or anything else.

6.3.6 direction

Syntax and usage:

```
<!ELEMENT direction (description?)>
<!ATTLIST direction
  flow (to-customer|from-customer|any) #REQUIRED >
```

Description:

Specifies the information flow direction.

If the direction is specified as 'to-customer' and the product is of the type 'query/reply' then this means that the query is made by `<principal>` to `<customer>` and that

2007-04-05

<customer> sends a reply back, and vice versa if the direction is 'from-customer'. Note that if the product is of the type 'query/reply' and 'any' has been specified then the query can be initiated by either principal or customer.

Elements:

description verbal description of the flow

Attributes:

flow	direction of the flow	
	to-customer	<principal> => <product> => <customer>
	from-customer	<principal> <=< <product> <=< <customer>
	any	<principal> <=< <product> <=< <customer> or <principal> => <product> => <customer>

6.3.7 confirm

Syntax and usage:

```
<!ELEMENT confirm EMPTY >
<!ATTLIST confirm
  required (yes|no) "yes">
```

Description

Describes when the delivery confirmation is used. Note that this is a confirmation from the addressee.

Attributes:

required	Specifies if the delivery confirmation is required to be used or not. When set to <i>no</i> a delivery confirmation may still be sent, but it is not required.
----------	--

6.3.8 billing

Syntax and usage:

```
<!ELEMENT billing ((per-exchange|per-volume|
  per-period),description?)>
<!ATTLIST billing
  required (yes|no) "yes"
  currency NMTOKEN "SEK">
```

2007-04-05

Description

Specifies how billing information should be collected, if at all. This information does not control how the SHS handles each message, but the SHS may use this information to control how it collects data to be used as a basis for invoice generation.

Elements:

per-exchange	invoicing is carried out for each exchange
per-volume	invoicing is carried out for each volume unit
per-period	invoicing is carried out for each period
description	a verbal description of the invoice

Attributes:

required	specifies whether or not invoices are to be used
currency	Specifies the currency of the invoice. The value is specified according to ISO 4217. As examples the value "SEK" is used for Swedish crowns and "DKK" is used for Danish crowns.

6.3.9 per-exchange

Syntax and usage:

```
<!ELEMENT per-exchange EMPTY>
<!ATTLIST per-exchange
  price NMTOKEN #REQUIRED >
```

Description

Specifies the price in the given currency per transfer.

Attributes:

price	specifies the price for a transfer with two decimal places. As a decimal symbol a comma is used.
-------	--

6.3.10 per-volume

Syntax and usage:

```
<!ELEMENT per-volume EMPTY >
<!ATTLIST per-volume
  price NMTOKEN #REQUIRED
  unit %SizeUnit; #REQUIRED >
```

2007-04-05

Description

Specifies the price in the given currency for the transfer of a given unit of volume.

Attributes:

<code>price</code>	specifies the price for the transfer as a floating point value with two decimal places.
<code>unit</code>	Specifies the unit for this volume price specification.

6.3.11 per-period

Syntax and usage:

```
<!ELEMENT per-period EMPTY >
<!ATTLIST per-period
  price NMTOKEN #REQUIRED
  unit %TimeUnit; #REQUIRED >
```

Description

Specifies the price in the given currency for a given period of time.

Attributes:

<code>price</code>	specifies the price for a period of time with two decimal places
<code>unit</code>	specifies the size of a time period

6.3.12 error

Syntax and usage:

```
<!ELEMENT error (#PCDATA) >
```

Description

This element contains information about how errors should be handled. The SHS is not required to use this information. The information in this element is mainly intended for human interpretation and is written for SHS administrators.

6.3.13 description

Syntax and usage:

```
<!ELEMENT description (#PCDATA) >
```

2007-04-05

Description

Verbal description in any language.

6.3.14 **general**

Syntax and usage:

```
<!ELEMENT general (description?, valid, schedule, QoS)>
```

Description

The element general contains general information for information exchange between two parts.

Elements:

description	description of the information exchange
valid	validation period
schedule	times for transfer
QoS	Describes the quality of service requirements at service level.

6.3.15 **valid**

Syntax and usage:

```
<!ELEMENT valid (valid-from, valid-to?)>
```

Description

Specifies validation period for the agreement.

Elements:

valid-from	The date from which the agreement becomes valid.
valid-to	The date when the agreement ends to be valid.

6.3.16 **valid-from**

Syntax and usage:

```
<!ELEMENT valid-from EMPTY >
<!ATTLIST valid-from
    date NMTOKEN #REQUIRED >
```

2007-04-05

Description

Specifies the date from which the agreement starts to be valid.

Attributes:

date	Agreement start date. The dates follow ISO 8601 extended format 'yyyy-mm-dd'.
------	---

6.3.17 valid-to

Syntax and usage:

```
<!ELEMENT valid-to EMPTY >
<!ATTLIST valid-to
    date NMTOKEN #REQUIRED >
```

Description

Specifies the last date that the agreement is valid.

Attributes:

date	Agreement end date. The dates follow ISO 8601 extended format 'yyyy-mm-dd'.
------	---

6.3.18 schedule

Syntax and usage:

```
<!ELEMENT schedule (intervaltime, starttime?,
    stoptime?)>
<!ATTLIST schedule
    startdate NMTOKEN #IMPLIED
    stopdate NMTOKEN #IMPLIED
    timezone CDATA #IMPLIED >
```

Description

Describes the date and time for planned transfers. This element is modelled on the Microsoft CDF (channel description format) standard.

This information is intended as a documentation of what has been agreed, and the SHS is not required to use this information.

Elements:

intervaltime	Specifies the time interval between transfers.
--------------	--

2007-04-05

<code>starttime</code>	Specifies the earliest transfer time for each twenty-four hours.
<code>stoptime</code>	Specifies the latest transfer time for each twenty-four hours.

Attributes:

<code>startdate</code>	Specifies when the planned transfers become valid. Must be greater than or equal to the date from which the agreement is valid. Default is the agreement start date. The date value follows ISO 8601 extended format 'yyyy-mm-dd'.
<code>stopdate</code>	Specifies when the planned transfers become invalid. Must be less than or equal to the date when the agreement ceases to be valid, including eventual extensions. Default is the agreement end date. The date value follows ISO 8601 extended format 'yyyy-mm-dd'.
<code>timezone</code>	Specifies the time zone according to ISO 8601.

6.3.19 intervaltime

Syntax and usage:

```
<!ELEMENT intervaltime EMPTY>
<!ATTLIST intervaltime %TimeQuantAttList;>
```

Description

Specifies the time interval between transfers.

Attributes:

<code>day</code>	Specifies the number of days in the time interval
<code>hour</code>	Specifies the number of hours in the time interval
<code>min</code>	Specifies the number of minutes in the time interval

The times in the attributes are added together, and if an attribute is missing it counts as zero. For example, `<intervaltime hour="2" min="30">` specifies the same time interval as `<intervaltime min="150">`.

6.3.20 starttime

Syntax and usage:

```
<!ELEMENT starttime (#PCDATA)>
```

Description

Specifies a start time. The time is specified in ISO 8601 extended format 'hh:mm'.

2007-04-05

6.3.21 **stoptime**

Syntax and usage:

```
<!ELEMENT stoptime (#PCDATA)>
```

Description

Specifies a stop time. The time is specified in ISO 8601 extended format 'hh:mm'.

6.3.22 **QoS**

Syntax and usage:

```
<!ELEMENT QoS (open, (volume, frequency)?, response?, description?)>
```

Description

Describes different qualities of service properties that apply. This information is included for informational purposes, and it is not required that a SHS uses this information in any way.

Elements:

open	specifies the open times for exchange
volume	specifies the volumes that can be exchanged
frequency	specifies the frequency of the information exchange
response	specifies the conditions for a requested answer concerning the exchange
description	verbal description of QoS

6.3.23 **open**

Syntax and usage:

```
<!ELEMENT open (when, (starttime, stoptime)*, description?)>
```

Description

Specifies open times during the day and during the week.

Elements:

when	Specifies which kind of days and part of days that the exchange is possible.
------	--

2007-04-05

starttime	The time from which exchange is possible for each twenty-four hours.
stoptime	The time until which exchange is possible for each twenty-four hours.
description	Verbal description of open times.

6.3.24 when

Syntax and usage:

```
<!ELEMENT   when          (description?)>
<!ATTLIST  when
  day          (every|work|weekend|public-holiday) "every"
  hours        (all|office|extended|night)         "all">
```

Description

Specifies days of the week and hours of the day.

Elements:

description	verbal description
-------------	--------------------

Attributes:

day	Can be every day, working days, weekends or holidays.
hours	Can be 24 hours, office hours, extended office hours or nights.

6.3.25 volume

Syntax and usage:

```
<!ELEMENT   volume        (average, peak?, description?) >
<!ATTLIST  volume
  per-transfer  NMTOKEN          #REQUIRED
  unit          %SizeUnit;      "KB">
```

Description

Describes the volumes during average load and peak load. The values specify the volumes of information transfer that have been agreed.

Elements:

average	average load
peak	peak load

2007-04-05

description verbal description

Attributes:

per-transfer The average data size that is exchanged in each transfer. Is specified as a whole number.

unit The unit for the data size specification.

6.3.26 average

Syntax and usage:

```
<!ELEMENT average EMPTY >
<!ATTLIST average
  number-per NMTOKEN #REQUIRED
  period %TimeUnit; #REQUIRED >
```

Description

Specifies the average number of exchanges per time period.

Attributes:

number-per Specifies the number of exchanges per time period.

period Specifies the time unit for a time period.

6.3.27 peak

Syntax and usage:

```
<!ELEMENT peak EMPTY >
<!ATTLIST peak
  number-per NMTOKEN #REQUIRED
  period %TimeUnit; #REQUIRED >
```

Description

Specifies the peak number of exchanges per time period.

Attributes:

number-per Specifies the number of exchanges per time period.

period Specifies the time unit for a time period.

2007-04-05

6.3.28 frequency

Syntax and usage:

```
<!ELEMENT frequency (average, peak?, description?) >
```

Description

Describes the average and peak frequency for information exchange.

Elements:

average	average load
peak	peak load
description	verbal description

6.3.29 response

Syntax and usage:

```
<!ELEMENT response ((request, reply?)?, description?) >
<!ATTLIST response
  within NMTOKEN #REQUIRED
  unit %TimeUnit; #REQUIRED >
```

Description

Describes when a reply is required after a query has been placed. This element can be empty, in which case the attributes for requesting an answer within a certain time period is used. If both 'request' and 'reply' have been specified then the element specifying the quickest answer is used.

Elements:

request	Specifies when a query can be placed.
reply	Specifies when a reply is required.
description	verbal description

Attributes:

within	Specifies the number of time units that are allowed to elapse before a response must be received by the caller.
unit	Specifies the time unit for the time specification in attribute <i>within</i> .

2007-04-05

6.3.30 request

Syntax and usage:

```
<!ELEMENT request (not-before, not-after?) >
```

Description

Describes when a query can be made.

Elements:

<code>not-before</code>	A query cannot be made before this time.
<code>not-after</code>	A query cannot be made after this time.

6.3.31 reply

Syntax and usage:

```
<!ELEMENT reply (not-before, not-after?) >
```

Description

Specifies the times for when a reply can be expected.

Elements:

<code>not-before</code>	A reply cannot be expected before this time.
<code>not-after</code>	A reply cannot be expected after this time.

6.3.32 not-before

Syntax and usage:

```
<!ELEMENT not-before (week-time) >
```

Elements:

<code>week-time</code>	Specifies the day of week and time of day from which current specification is valid.
------------------------	--

2007-04-05

6.3.33 not-after

Syntax and usage:

```
<!ELEMENT not-after (week-time) >
```

Elements:

<code>week-time</code>	Specifies the day of week and time of day until which current specification is valid.
------------------------	---

6.3.34 week-time

Syntax and usage:

```
<!ELEMENT week-time EMPTY >
<!ATTLIST week-time
  day %DaySpec; "every"
  time NMTOKEN #IMPLIED >
```

Description

Specifies a day of the week and optionally also a time. Also all days of the week may be specified. The time is to be associated with the specified day.

Attributes:

<code>day</code>	Specifies a weekday or all weekdays.
<code>time</code>	Specifies the time of a day following ISO 8601 extended format 'hh:mm'. If omitted the value is interpreted as '00:00'.

2007-04-05

7 Product

7.1 Introduction

This DTD describes how a product type XML description is composed. The product type is a key definition of information exchanged by SHS users. The product type governs how application information (data part) is exchanged. Examples of information that the product type defines:

- Owner of a specific information definition (principal)
- Instructions to users e.g. if response is required for a specific product type
- Unique identities

7.2 Entities

7.2.1 HrefAtt

Syntax and usage:

```
<!ENTITY % HrefAtt 'CDATA #IMPLIED'>
```

This entity is used to restrict the URI 'pointer' to external references.

7.3 Elements

7.3.1 shs.product

Syntax and usage:

```
<!ELEMENT shs.product (principal,data+,reply-data*,description?)>
<!ATTLIST shs.product
  version CDATA #FIXED "1.2"
  uuid NMTOKEN #REQUIRED
  commonName CDATA #IMPLIED
  resp-required (yes|no) "yes"
  labeledURI %HrefAtt; >
```

Description

A product type is described by

- Principal
- One or several primary types of data parts
- Zero, one or several reply types of data parts
- Description

2007-04-05

Note that XML files that are based on this DTD describe meta-data for the way the messages (products) are constructed as data structures that are used for information transfer. The information transport is HTTPS transfer of MIME and S/MIME messages.

This description can be seen as if the SHS can read a product type definition and generate the correct MIME structure from it. The receiver, who has the same product type definition, will also be able to see what is contained in the incoming MIME structure and how signing and encryption is to be handled.

The model for this DTD describes a protocol between business applications. The product type definition is a definition of information exchange at business level. A business initiates a transfer by sending a SHS message that in turn can be made up of one or several data parts. The receiving business system can answer by sending a reply if it is a 'query/reply' relation.

[Business System 1] — <data>+ → [Business System 2]

or

[Business System 1] — <data>+ → [Business System 2]
[Business System 1] ← <reply-data>+ — [Business System 2]

All these relations form a product type, that is, a product type is a description of a very simple protocol and the information content of the exchange.

The SHS special product types are not instantiated or specified like the other normal product types. In agreements, addresses and labels these special product types may be associated with a *commonName* together with the ID. For these special product types SHS specifies the following common names:

<u>ID</u>	<u>CommonName</u>
confirm	Delivery confirmation message
error	Error message
agreement	Agreement exchange message

When sending these special product types the label attribute *sequence-type* must be set to *adm*. The two special product types *confirm* and *error* are described in chapter 8. The special product type *agreement* must always contain exactly one data part that is an xml-document according to the *shs.agreement* DTD specification

Elements:

<code>principal</code>	The owner of the product type.
<code>data</code>	data that initiates the information exchange (one or several)

2007-04-05

reply-data data that forms the reply in a query/reply relation
description verbal description

Attributes:

version Specifies the version of SHS product type dtd that an instance of the shs.product root element conforms to. This attribute is used together with the DOCTYPE specification of a product type document??!! to decide the version of dtd to use when reading and parsing the shs.product element.

uuid The identifying uuid for the product type.

commonName Common name for the product, for example 'RSV.Taxering.Näringsverksamhet.Skatteåterbetalning.2001'. The value is preferably the same as the *productName* in the SHS product type catalogue.

resp-required specifies whether an answer is required

labeledURI URI for more information about the product type. The value is probably the same as the *labeledURI* in the SHS product type catalogue.

7.3.2 principal

Syntax and usage:

```
<!ELEMENT principal (#PCDATA) >
<!ATTLIST principal
  commonName CDATA #IMPLIED
  labeledURI %HrefAtt; >
```

Description

Specifies identity and 'pointer' to more information about the principal. The principal 'owns' the product type. The element value contains a unique identity for the owner of the product type. The <shs-actor> production in chapter 4 defines the syntax for the element value.

Attributes:

commonName Common name for <principal>, found in the SHS catalogue. The value is preferably the same as the *organizationName* in the actor catalogue.

labeledURI URI that points to further information about <principal> that can be found in the SHS catalogue.

2007-04-05

7.3.3 data

Syntax and usage:

```
<!ELEMENT data (description?, mime, security?) >
<!ATTLIST data
  datapartType CDATA #REQUIRED
  minOccurs NMTOKEN "1"
  maxOccurs NMTOKEN "1"
  data-type %HrefAtt; >
```

Description

Specifies one type of data part that is handled by this product type.

The data part type attribute is copied to the message label when sending messages and from the label to the message information when listing messages. This gives a reference between the corresponding data part specification and a specific data part in a message.

Elements:

description	verbal description
mime	describes how MIME is used
security	describes the security level for each individual data part

Attributes:

datapartType	Specifies the type of the data part. Should be short and distinct. Is intended for automatic computer reading and interpretation.
minOccurs	Defines the minimum number of occurrences for this data part. The value must be zero or a positive integer that is less than or equal to <i>maxOccurs</i> .
maxOccurs	Defines the maximum number of occurrences for this data part. The value must be minus one (-1) or a positive integer that always is greater than 0 and is greater than or equal to <i>minOccurs</i> . The minus one value means that the maximum number of occurrences is unlimited.
data-type	pointer to an external description of the data part type that can be a DTD file or some other description. If there is no DTD then this could either be machine-readable grammar that is not a DTD or a simple verbal description.

7.3.4 description

Syntax and usage:

```
<!ELEMENT description (#PCDATA) >
```

2007-04-05

Description

Specifies a description. The value is a text field intended for a description.

7.3.5 mime

Syntax and usage:

```
<!ELEMENT mime EMPTY >
<!ATTLIST mime
  type NMTOKEN "text"
  subtype NMTOKEN "xml"
  text-charset NMTOKEN "iso-8859-1"
  transfer-encoding (binary|base64) "binary" >
```

Description

Describes how MIME is used to code the product.

The *type* and *subtype* attributes define the content type specification in the MIME header (“Content-Type: *type/subtype*”).

Attributes:

<code>type</code>	specifies the MIME type. All IANA types are allowed.
<code>subtype</code>	specifies the MIME subtype. All IANA subtypes are allowed.
<code>text-charset</code>	Specifies the character set that is to be used. This attribute is only significant when the MIME type attribute is set to “text”.
<code>transfer-encoding</code>	Specifies the transfer encoding that is to be used when a SHS message containing this product type is sent.

7.3.6 security

Syntax and usage:

```
<!ELEMENT security (digest?, encryption?, dsig?)>
```

Description

Describes the security requirements at message level (note that a SHS message can consist of several separate messages).

Elements:

<code>digest</code>	describes the condensate requirements
<code>encryption</code>	describes the encryption requirements
<code>dsig</code>	describes the requirements for digital signatures

2007-04-05

7.3.7 digest

Syntax and usage:

```
<!ELEMENT digest EMPTY >
<!ATTLIST digest
  algorithm (SHA1|MD5) "SHA1" >
```

Description

This element defines the digest algorithm used for digital signatures.

Attributes:

algorithm The algorithm name.

7.3.8 encryption

Syntax and usage:

```
<!ELEMENT encryption EMPTY >
<!ATTLIST encryption
  algorithm (RC4|3DES) "RC4"
  key-length (56|128) "128" >
```

Description

Specifies whether or not the data part is to be encrypted.

Attributes:

algorithm The cryptographic algorithm to use. The two algorithms that are available are RC4 and 3DES.

key-length key length for encryption. This can be ignored when the algorithm name has an implied key length.

7.3.9 dsig

Syntax and usage:

```
<!ELEMENT dsig EMPTY >
<!ATTLIST dsig
  algorithm CDATA #FIXED "RSA"
  key-length (512|1024|1536|2048) "1024" >
```

Description

Specifies whether or not the data part is to be digitally signed.

2007-04-05

Attributes:

algorithm	The cryptographic algorithm to use. The only used algorithm is RSA.
key-length	key length for the cryptographic algorithm. This can be ignored when the algorithm name has an implied key length.

7.3.10 reply-data

Syntax and usage:

```
<!ELEMENT reply-data (description?, mime, security?) >
<!ATTLIST reply-data
  datapartType CDATA #REQUIRED
  minOccurs NMTOKEN "1"
  maxOccurs NMTOKEN "1"
  data-type %HrefAtt; >
```

Description

This element is used to describe the design of a data part that is part of a reply to an earlier query in a query/reply relation.

The reply data part type attribute is copied to the message label when sending reply messages and from the label to the message information when listing messages. This gives a reference between the corresponding reply data part specification and a specific data part in a reply message.

Elements:

description	verbal description
mime	describes how MIME is used
security	describes the security level for each individual message

Attributes:

datapartType	Specifies the type of the data part. Should be short and distinct. Is intended for automatic computer reading and interpretation.
minOccurs	Defines the minimum number of occurrences for this data part. The value must be zero or a positive integer that is less than or equal to <i>maxOccurs</i> .
maxOccurs	Defines the maximum number of occurrences for this data part. The value must be minus one (-1) or a positive integer that always is greater than 0 and is greater than or equal to <i>minOccurs</i> . The minus one value means that the maximum number of occurrences is unlimited.

2007-04-05

data-type pointer to an external description of the data part type that can be a DTD file or some other description. If there is no DTD then this could either be machine-readable grammar that is not a DTD or a simple verbal description.

2007-04-05

8 Management

The administrative messages that are sent in SHS messages fall into two categories:

- delivery confirmation
- error messages

XML documents are used for common processing with a common SHS management dtd. These XML documents are sent together with an SHS label as an SHS message.

8.1 Synchronous calls

For synchronous calls the delivery confirmations and error messages play a double role. A delivery confirmation is used for cases where a synchronous call does not send any application data in return, which can occasionally also be interpreted as a processing confirmation since it is not sent until the call to the receiving application system is ready. The error message mechanism can be used when a synchronous call to an application system fails.

It is possible to send further information, which can be useful for finding out exactly what went wrong with a synchronous call to a business system, among other things. The element *appinfo* is used for such information.

8.2 Error Codes

The error codes that can be found in the `errorcode` attribute is defined in SHS protocol specification [Protocols] chapter 8.

8.3 Elements

8.3.1 shs.management

Syntax and usage:

```
<!ELEMENT shs.management (datetime, (confirmation|error)) >
<!ATTLIST shs.management
  version CDATA #FIXED "1.2"
  corr.id NMTOKEN #REQUIRED
  content.id NMTOKEN #REQUIRED>
```

Description

This is the top-level element in a management message. It references the original message through the *corr.id* attribute. It can contain either a confirmation or an error element, which indicates the type of management information. If it contains a confirmation element, this is a delivery confirmation message, and if it contains an error

2007-04-05

element, this is an error message. The product type element in the label must also be set one of the special product type values *confirm* or *error*.

Elements:

<code>datetime</code>	The date and time stamp of the message content.
<code>confirm</code>	Message delivery confirmation. The confirmation is sent when the receiver has fetched the message.
<code>error</code>	Message delivery error. The error is sent when a requested message can not be delivered as requested.

Attributes:

<code>version</code>	Specifies the version of SHS management dtd that an instance of the <code>shs.management</code> root element conforms to. This attribute is used together with the DOCTYPE specification of a management message to decide the version of dtd to use when reading and parsing the <code>shs.management</code> element.
<code>corr.id</code>	The correlation ID taken from the original message.
<code>content.id</code>	The content ID taken from the original message.

8.3.2 **datetime**

Syntax and usage:

```
<!ELEMENT datetime (#PCDATA) >
```

Description

This is a timestamp. It contains date and time information about when this message content was created or assembled. The timestamp is formatted according to ISO 8601 extended format 'yyyy-mm-ddThh:mm:ss'.

8.3.3 **confirmation**

Syntax and usage:

```
<!ELEMENT confirmation (appinfo*) >
```

Description

This element is used to indicate a delivery confirmation. It can contain extra, application-specific, information in `appinfo` elements.

2007-04-05

Elements:

appinfo Application specific information.

8.3.4 error

Syntax and usage:

```
<!ELEMENT error (appinfo*) >
<!ATTLIST error
  errorcode NMTOKEN #REQUIRED
  errorinfo CDATA #IMPLIED >
```

Description

This element is used to indicate a delivery error, and uses an error code to identify the error. It can contain extra, application-specific, information in appinfo elements.

Elements:

appinfo Application specific information.

Attributes:

errorcode An error code. Error codes for SHS is specified in the protocol specification [Protocol] chapter 8.

errorinfo A textual description of the error. This can be used to present the error to humans.

8.3.5 appinfo

Syntax and usage:

```
<!ELEMENT appinfo (#PCDATA) >
<!ATTLIST appinfo
  name NMTOKEN #REQUIRED >
```

Description

This element is used for application-specific information. It serves as a generic means to add information to the management messages, by allowing free-form named data blocks.

The contents of this element are an arbitrary string.

Attributes

name The name of this data block.

2007-04-05

8.4 Examples

Example 1

```
<?xml version="1.0"?>
<!DOCTYPE shs.management SYSTEM "shs-management-1.2.dtd">

<shs.management corr.id="Tandred-Got-1234-0208/1"
  content.id="Tandred-Got-1234-0208">
  <datetime>1999-04-27T11:40:00</datetime>
  <confirmation/>
</shs.management>
```

Example 2

```
<?xml version="1.0"?>
<!DOCTYPE shs.management SYSTEM "shs-management-1.2.dtd">

<shs.management corr.id="RFV-VS-AKT-200011-071/2"
  content.id="RFV-VS-AKT-200011-071">
  <datetime>1999-04-27T11:40:00</datetime>
  <error errorcode="UnknownReceiver"
    errorinfo="Receiving actor unknown">
    <appinfo name="Receiver">urn:X-shs:2021000000</appinfo>
  </error>
</shs.management>
```

2007-04-05

9 Message lists

This DTD is used to generate the message list sent as a response to an HTTP request for messages available for fetching.

The list may either contain all messages that a specified receiver has available for fetching, or a subset of such available messages. The message list request uses a number of different parameters to define which messages and elements to include in the list. These parameters are described in [Protocols]

9.1 Elements

9.1.1 shs.message-list

Syntax and usage:

```
<!ELEMENT shs.message-list (message*) >
<!ATTLIST shs.message-list
  version CDATA #FIXED "1.2">
```

Description

This is the top-level element. It contains a list of message elements.

Elements:

<code>message</code>	Message available for fetching and satisfying specified listing parameters.
----------------------	---

Attributes:

<code>version</code>	Specifies the version of SHS message list dtd that an instance of the shs.message-list root element conforms to. This attribute is used together with the DOCTYPE specification of a message list to decide the version of dtd to use when reading and parsing the shs.message-list element.
----------------------	--

2007-04-05

9.1.2 message

Syntax and usage:

```
<!ELEMENT message (meta*, subject?, data*) >
<!ATTLIST message
  tx.id          NMTOKEN          #REQUIRED
  timestamp      CDATA            #REQUIRED
  corr.id        CDATA            #IMPLIED
  content.id     CDATA            #IMPLIED
  size           CDATA            #IMPLIED
  originator     CDATA            #IMPLIED
  from           CDATA            #IMPLIED
  to             CDATA            #IMPLIED
  end-recipient  CDATA            #IMPLIED
  product        CDATA            #IMPLIED
  sequence-type (event|request|reply|adm) #REQUIRED
  status         (production|test) "production" >
```

Description

This element describes one message that is available for fetching.

Many of the attribute values for this element are copied from the corresponding attribute in the message label.

Elements:

meta	Message content annotations.
subject	Short description of message content.
data	Information about the different data parts that are contained within the message.

Attributes:

tx.id	The transaction ID of the message
timestamp	A timestamp that indicates when the message became ready to fetch. The timestamp is formatted according to ISO 8601 extended format 'yyyy-mm-ddThh:mm:ss'.
corr.id	The correlation ID of the message
size	The size of the message. It specifies the total number of bytes that the message consists of. This also includes the sizes of the MIME headers and the SHS label.
originator	The end user that created the message content from the beginning.
from	The address of the sender. The <shs-address> production in chapter 4 defines the syntax for the attribute value.

2007-04-05

to	The address of the receiver. The <shs-address> production in chapter 4 defines the syntax for the attribute value.
end-recipient	The end user that is the final recipient of the message.
product	A unique identity for the product type of the message. The <shs-product> production in chapter 4 defines the syntax for the attribute value.
sequence-type	The sequence type of the message
status	The status of the message. It is set to <i>production</i> for normal and real production and to <i>test</i> for production test.

9.1.3 meta

Syntax and usage:

```
<!ELEMENT meta (#PCDATA) >
<!ATTLIST meta
  name CDATA #REQUIRED >
```

Description

This element makes it possible to create a annotation for the message content. It consists of a name-value pair. This element is intended to be a short non-verbal annotation or description and to be used mostly for automatic computer reading and interpretation. This information is application specific and has no specified control function for the SHS.

This element can be used to implement content dependent decisions. If desirable and needed, it can also be used to create different sort of subgroups within one product type.

For a specific name the value of this element often will be chosen from a list of enumerated possible values. The different parts that are handling this element will agree upon this enumerated list in advance.

The element value may contain any form of information that is agreed upon by the different parts that are handling this element.

Attributes:

name	The name of the content annotation.
------	-------------------------------------

9.1.4 subject

Syntax and usage:

```
<!ELEMENT subject (#PCDATA) >
```

2007-04-05

Description

This element contains a description of the message content. This is a verbal description intended for human reading. This information is application specific and has no control function for the SHS.

The element may contain any form of information that is descriptive for the content of the message.

9.1.5 data

Syntax and usage:

```
<!ELEMENT data EMPTY >
<!ATTLIST data
  datapartType CDATA #REQUIRED
  filename CDATA #IMPLIED
  no-of-bytes NMTOKEN #IMPLIED
  no-of-records NMTOKEN #IMPLIED >
```

Description

This element contains information about one data part of the message. Many of the attribute values for this element are copied from the corresponding attributes in the message label.

It is strongly recommended that a SHS supply as much as possible of this information. A user or client may use this information to decide whether to fetch a message or not.

Attributes:

<code>datapartType</code>	Specifies the type of data part for this data part. Is intended for automatic computer reading and interpretation.
<code>filename</code>	A suggested name to use when retrieving and storing this data part into a local file. This could be the same name as when the data part was sent or it could be a new name that is recommended to use. The usage of this filename is optional and the user may choose to use another name even when the attribute is specified.
<code>no-of-bytes</code>	Specifies the size of the data part. It specifies the number of bytes that the MIME decoded content of this data part consists of. This size is without the size of any MIME headers.
<code>no-of-records</code>	Specifies the number of records. If a record structure is used, this attribute specifies the total number of records that are contained in the data part.

2007-04-05

9.2 Examples

Example 1

```
<?xml version="1.0"?>
<!DOCTYPE shs.message-list SYSTEM "shs-message-list-1.2.01.dtd">
<shs.message-list>
  <message tx.id="a5268ffe-fc0b-11d2-802d-0060b0836211"
    timestamp="2000-11-27T11:40:00"
    corr.id="RFV-VS-AKT-200011-071/2"
    content.id="RFV-VS-AKT-200011-071"
    size="11723"
    from="urn:X-shs:2021000548"
    to="urn:X-shs:2021000985.Taxering"
    product="urn:X-shs:a9268ffe-fc0b-11d2-802d-0060b0836299"
    sequence-type="request"
    status="production" >
    <meta name="region">gotland</meta>
    <meta name="kategori">tandl</meta>
    <subject>Taxeringsredovisning av tandläkare på Gotland för November 2000,
      från AKT/RFV</subject>
    <data datapartType="TaxRedData"
      filename="tax0011_09t1.trd"
      no-of-bytes="10235"
      no-of-records="189" />
    <data datapartType="TaxRedSum"
      filename="tax0011_09t1.trs"
      no-of-bytes="148"
      no-of-records="3" />
  </message>
  <message tx.id="b5268ffe-fc0b-11d2-802d-0060b0836211"
    timestamp="2000-11-26T23:17:00"
    corr.id="PRV-VS-KTN-200011-321/1"
    content.id="PRV-VS-KTN-200011-321"
    size="52531"
    from="urn:X-shs:2021000123"
    to="urn:X-shs:2021000985.Företag"
    product="urn:X-shs:b9268ffe-fc0b-11d2-802d-0060b0836299"
    sequence-type="event" >
    <meta name="period">2000-11</meta>
    <meta name="info">nystartf</meta>
    <subject>Information om nystartade företag November 2000</subject>
    <data datapartType="NystartFöretag"
      no-of-bytes="51235" />
  </message>
</shs.message-list>
```