# SHS Version 1.2.01 Architecture

**Verva - Swedish Administrative Development Agency**

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

# Contents

# 1 Introduction

This document describes the architecture of SHS. It intends to give the reader an overview of the functionality of SHS, its scope and architectural constraints. A more detailed description can be found in the references listed in the reference section below.

## 1.1 Audience

The primary target of this document is IT-architects, infrastructure specialists, programmers or really anyone who has a need to learn what SHS is about. This document is a prerequisite when reading the more detailed specifications referred to in the reference section below.

## 1.2 References

| | |
|---|---|
| [Protocols] | SHS Version 1.2 Protocols |
| [DTD] | SHS Version 1.2 DTD Descriptions |
| [API] | SHS Version 1.2 Application Interfaces |
| [DIR] | SHS Version 1.2 Directory |
| [CA] | SHS Version 1.2 CA |

## 1.3 Document history

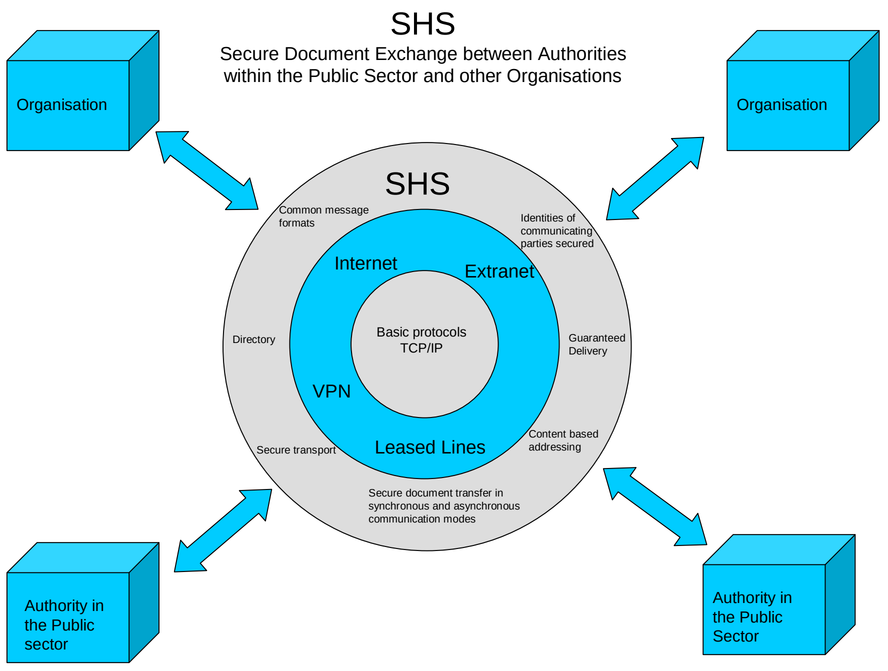
| Version | Date | Change | By | Approved |
|---|---|---|---|---|
| 0.1 | 2003-03-27 | Initial structure | Anders Bremsjö | |
| 0.2 | 2003-04-30 | Version to be reviewed in Luleå workshop | Anders Bremsjö | |
| 0.3 | 2003-05-27 | Version to be reviewed by reference group | Anders Bremsjö | |
| 1.2 | 2003-10-09 | Final version 1.2 | Anders Lindgren | Jan Lundh |
| 1.2 | 2006-04-25 | Document reorganisation to ovner of Verva | Christer Marklund | |

# 2 Architecture overview

## 2.1 Introduction

SHS is an infrastructure for information exchange primarily between authorities in the public sector, but also between authorities and enterprises.

The purpose has been to facilitate easy access, with high security based on standard protocols to enable secure information exchange across Internet (and Extranets, VPN, Leased Lines) at a reasonable cost. The purpose is also to make the architecture extensible to accommodate information exchange between Swedish citizens and public authorities. However, the SHS specifications do not cover the communication infrastructure to provide for the "last mile" connectivity to the citizens. This communication will be the concern of the individual authorities when offering services to the citizens.


SHS
Secure Document Exchange between Authorities within the Public Sector and other Organisations
Organisation
Organisation
SHS
Common message formats
Identities of communicating parties secured
Internet
Extranet
Basic protocols TCP/IP
Directory
Guaranteed Delivery
VPN
Leased Lines
Secure transport
Content based addressing
Secure document transfer in synchronous and asynchronous communication modes
Authority in the Public sector
Authority in the Public Sector

## 2.2 SHS design criteria's

The design of SHS is based on the following fundamental criteria's:

- Achieve loosely coupled systems.

- Secure end-to-end transmission of information using standards based technology for cryptography, supporting both signatures and encryption.
- Based on well-established standards with a broad industry acceptance.
- Designed with openness in mind to allow the adoption of evolving standards.
- Based on a distributed architecture with direct communication between sender and receiver/responder
- Support for multiple exchange patterns (synchronous, asynchronous, request/reply, one to many, many to one)
- Controlled communication flows by the use of agreements and predefined information structures (i.e. product types).
- Scalability to support for high volume transfers
- Support of large data transfers, that is multi Gbyte /message.

## 2.3 Definition of SHS terms

**SHS Messaging Service** - the component that executes the core services of SHS, which are routing, error handling logging etc.

**SHS Node** - an actual server implementation of the SHS messaging service.

**SHS Network** – the collection of SHS nodes that are interconnected in a network mesh.

**SHS Actor** – The SHS Actor is a part exchanging SHS messages with other actors. The actor has an SHS node (or accessing an SHS-node as a service) and has connected one or more of its business applications to the SHS network.

**SHS Product Type** – well defined structures of the artefacts transported in the SHS network, mainly various business documents. There is always an SHS actor who is the owner of a product type.

**SHS Product** – is an instantiated product type. One ore more products are packaged in an SHS message before submitted to SHS for transportation.

**SHS Message** – The structure SHS uses to exchange information. An SHS message can contain one or more SHS Products.

**SHS Data Part –** The smallest entity of information to be transferred by SHS. SHS data parts are packaged in an SHS message according to the product type definition.

**SHS Agreement –** a set of rules that governs the handling of a given product type between two actors. An agreement declares for example transfer cost, confirmation requirements and if a product is composed of a reply/request scenario. The principal (one of the actors) is the owner of the agreement.

**SHS Address –** the means by which a communicating party is referenced

**SHS Directory –** a globally available repository where the SHS messaging service may find information on other SHS Actors, SHS Products, SHS addresses and public agreements.

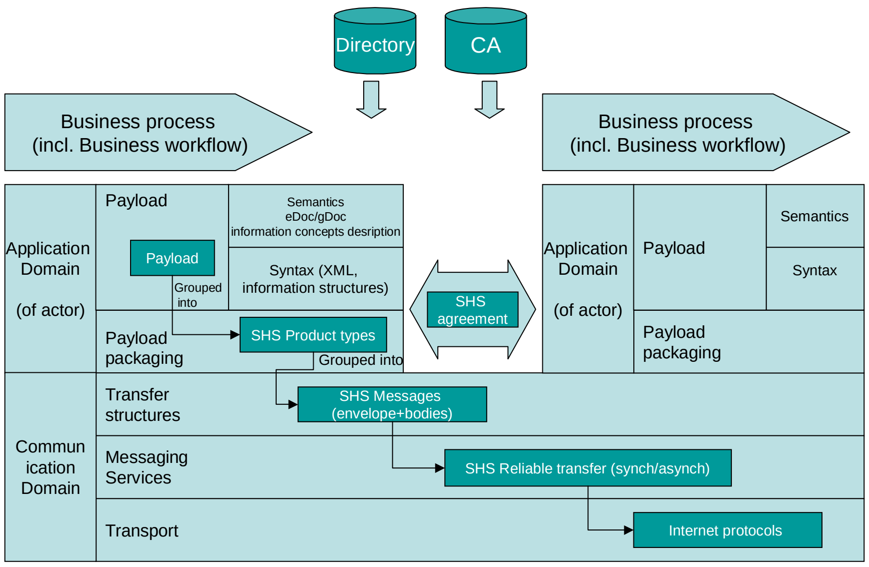
**SHS Interface –** methods that define how a messaging service can be utilized by an application or other service. The interface is implemented by a number of API's and client applications

**Business system –** any type of application that may need to exchange information with other systems or organisations, e.g. a tax administration system. The business system connects to SHS using one of the interfaces to SHS.

**CA –** the Certificate Authority is an authority that issues and manages security credentials and public keys. SHS uses external providers of these services.

## 2.4 SHS scope

This section intends to put SHS into context and describe what SHS specifications cover and do not cover. SHS consists of basic and extended services. The following chapters primarily describe the basic services, expect chapter 11 where extended services are described. The figure below depicts a general description "stack" covering layers from the basic transport all the way up to the business processes. SHS properties are highlighted in the different layers of this description "stack".


Directory
CA
Business process (incl. Business workflow)
Business process (incl. Business workflow)
Application Domain (of actor)
Payload
Semantics eDoc/gDoc information concepts desription
Payload
Grouped into
Syntax (XML, information structures)
Payload packaging
SHS Product types
Grouped into
SHS agreement
Application Domain (of actor)
Payload
Semantics
Syntax
Payload packaging
Communication Domain
Transfer structures
SHS Messages (envelope+bodies)
Messaging Services
SHS Reliable transfer (synch/asynch)
Transport
Internet protocols

The SHS specifications include:

- Definitions of how application payload (business document) is packaged using SHS Product types with possibilities to secure the payload with signatures and encryption.
- Definitions of how information structures and communication patterns for an SHS product type are exchanged in an agreement between actors.
- Definitions of how SHS actors can locate services using the SHS-directory.
- Definitions of the transfer structure using SHS messages.
- Definitions of how information is routed and reliably transferred using the SHS Messaging Service (SHS MS).
- Defines how the SHS messages are transported securely (authentication and encryption) by internet protocols.

SHS does not:

- Define business semantics
- Provide for business system level security

- Provide any other application capability on top of packaging and transferring. However, SHS is extensible and can be used in conjunction with software that provides for business workflow, conversion, formatting etc. See section 11 on extended services.

## 2.5 Architecture perspectives

The SHS architecture is described using four different perspectives:

1. **The service perspective** describes a service view of SHS. This view describes the relations between the communicating parties, agreements, product types and directory. It also describes general communication patterns between communicating parties.

2. **The protocol perspective** describes the different protocol layers, or the “wire stack”, and the interfaces to these layers.

3. **The network perspective** describes the different components of SHS in a network context.

4. **The security perspective** describes the security mechanisms within SHS.

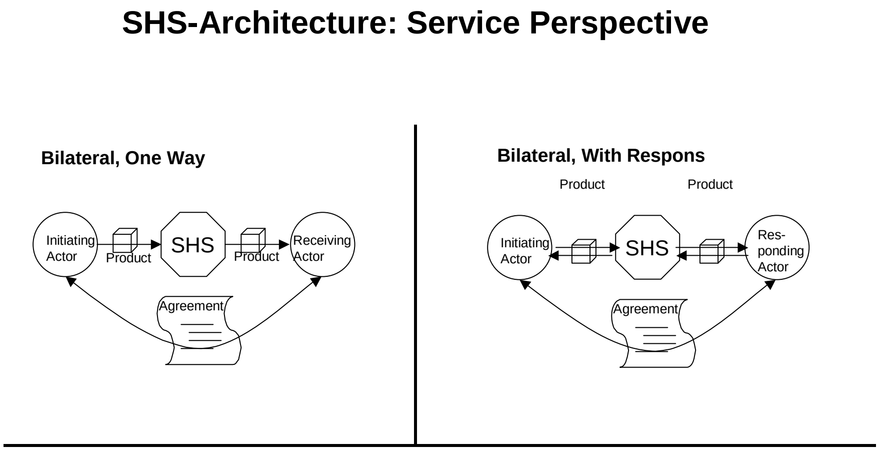
### 2.5.1 The service perspective

SHS provides for a message-oriented architecture where messages are transported from an initiator to a receiver/responder across the SHS Messaging Service. There are three communications patterns in the use of SHS services. All of these support asynchronous and synchronous communication. See chapter 4 for a deeper description on the asynchronous/synchronous functionality.

The figures below describes the three basic communication patterns. Note that in the figures below the SHS symbol denotes the SHS functionality as a whole and can constitute one or two SHS-nodes involved in the transfer. The network perspective describes this and is given in section 2.5.3.

Communication patterns described:

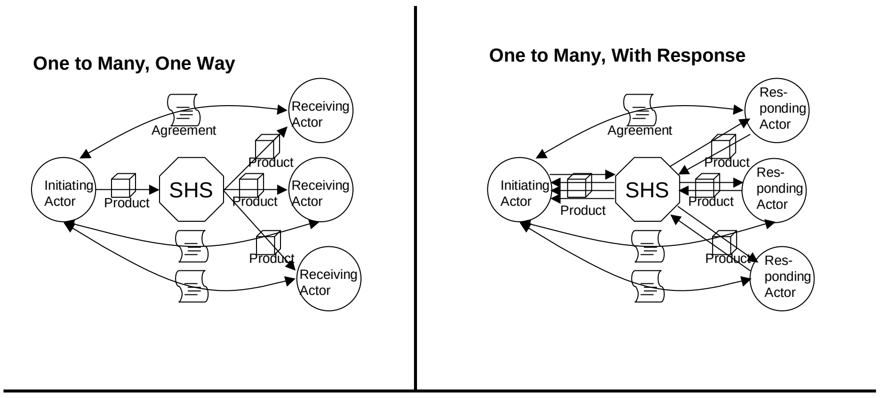
1. **Bilateral communication – Communication between two parties**



a. **Bilateral, one-way communication**. This pattern is used when an organisation needs to deliver information (defined by a product type) to a receiver where no response is needed (apart from confirmations of delivery). This pattern can use either asynchronous or synchronous communication. An SHS agreement defines the bilateral communication and what product types that can be communicated. Either the initiating actor or the receiving actor can be the principal of the agreement.

b. **Bilateral, with response.** This pattern describes a request/response scenario. The initiating party delivers an SHS message to the receiver who in turn responds with an SHS message. This pattern can use either asynchronous or synchronous communication. An SHS agreement defines the bilateral communication and what product types that can be communicated. The request and the related response messages are the transport packaging of a product (defined by a product type). Either the initiating actor or the responding actor can be the principal of the agreement.

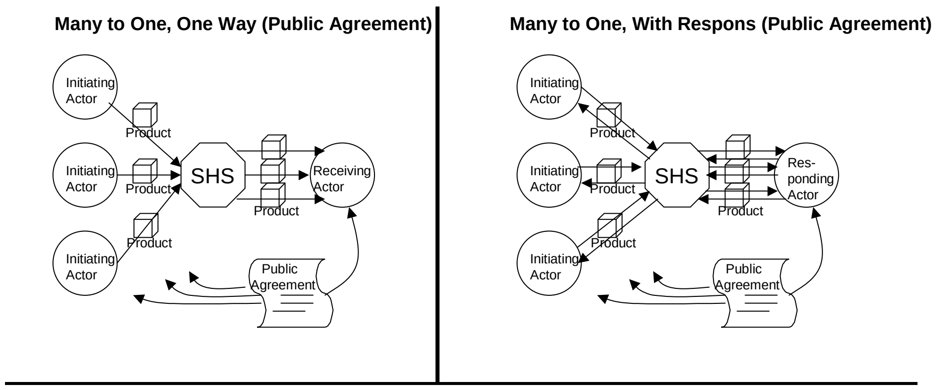
2. **One to Many.**



a. **One to Many, One Way.** This pattern is used when there is a need to send the same information (defined by the product type) to many receivers. This is done by establishing an agreement with the initiator and all the receiving parties. The initiator is always the principal of the agreement. This pattern can use only the asynchronous mode of communication.

b. **One to Many, With Response.** Same as above with the addition that the receiving actors can send a response.

3. **Many to One (Public Agreement)**



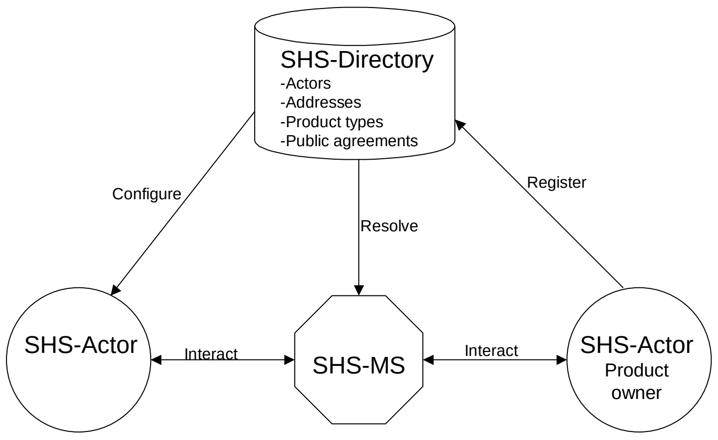
a. **Many to One, One Way (Public agreement).** This pattern is used when a number of organisations need to send the same type of information (defined by a product type) to one receiver. This is done by declaring a public agreement with the receiver as principal. This pattern can use both synchronous and asynchronous mode of communication.

b. **Many to One, With Response (Public agreement).** Same as above with the addition that the receiver sends a response to the originator.

#### *2.5.1.1 Use of directory*

All of the described communication patterns use the same method of locating services and finding addresses. The figure below describes the role of the directory in the establishment of communication.

SHS-Architecture: Use of directory

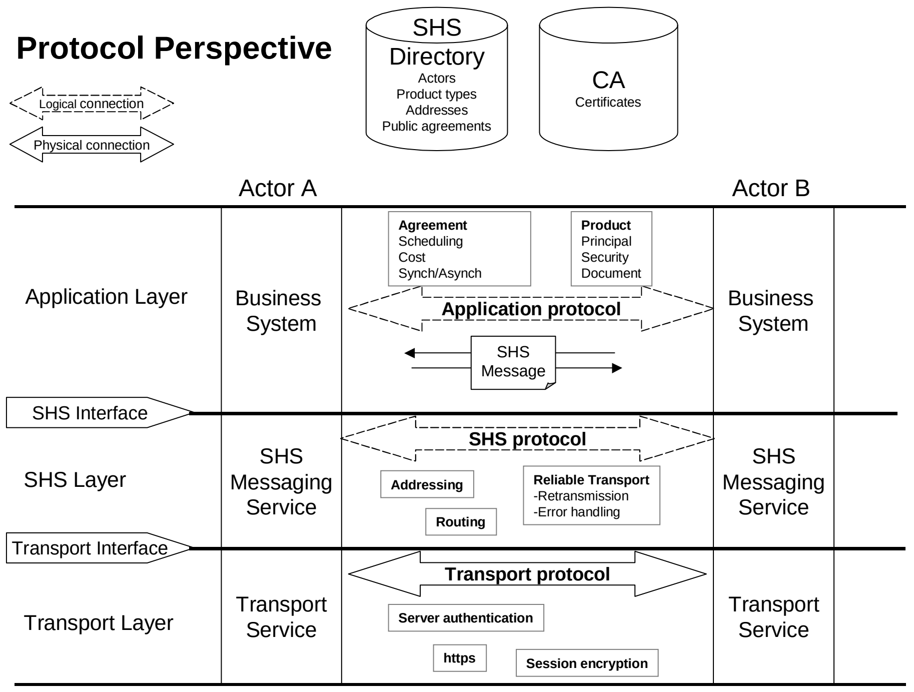


The product owner registers its organisation, address and product types owned by the organisation. Public agreements are also stored in the directory if a product type is used in a many to one service perspective. The business applications may use the directory information at configuration time to establish communication parameters. The SHS messaging service uses the directory to resolve product and organisation information to eventually locate the physical address of the communicating party.

### 2.5.2 Protocol perspective

SHS defines three layers of protocols. These are the Application layer, the SHS layer and the Transport layer.

The figure below illustrates the different layers.



The protocol interfaces separates the protocol layers and define how to access the services of a particular layer. Only the interface to the SHS layer is fully defined in the SHS specifications.

The SHS Directory provides a global repository for information about actors, products, addresses and public agreements.

Certificates for server authentication, encryption and document signing are provided from an external Certificate Authority (CA).

#### *2.5.2.1 Application layer*

The application layer defines the interaction between two business systems. SHS specifies some parts of this interaction using the definitions of primarily the SHS Agreement and to some extent the SHS Product. The SHS Product specifies the information document structure, its security level (signed, encrypted) and in general terms the communication sequence (if a reply is requested as a result of an information transfer). The SHS Agreement specifies more detailed information on the particular bilateral exchange between two business applications. Examples are intervals of exchange, volumes, billing, handling of confirmation and communication mode (synchronous or asynchronous).

The application layer interface is not defined but will constitute the creation and interpretation of SHS messages, which can contain one or more SHS products. It will also constitute the exchange sequence between two applications.

#### *2.5.2.2 SHS Layer*

The SHS layer provides:

- Delivery using either a store and forward mechanism (asynchronous mode).
- Immediate delivery and response (synchronous mode).
- Delivery confirmations
- Routing of SHS messages including forwarding to multiple recipients and subscriptions.
- Exchange of agreements.
- Error handling and the creation of log entries.

The SHS interface is defined in three forms:

1. Formats and Protocols: An application can interface the SHS using standard formats and protocols defined in the specifications.
2. External programs that can be invoked from the business application. These handle basic interaction with SHS.
3. API – Application Programming Interface. The business application will use the API to tightly connect to the SHS. The API is of today defined in:
    a. A pseudo format API, which does not guarantee code portability between different vendors' implementation of the API (This was the basis for the initial c-API's supplied by the vendors).
    b. Java API
    c. Web services API

End-to-end message encryption and message signing using the SHS certificates are supported in both the external program and API implementation. For more information on SHS interfaces, see chapter 3 and references [API].

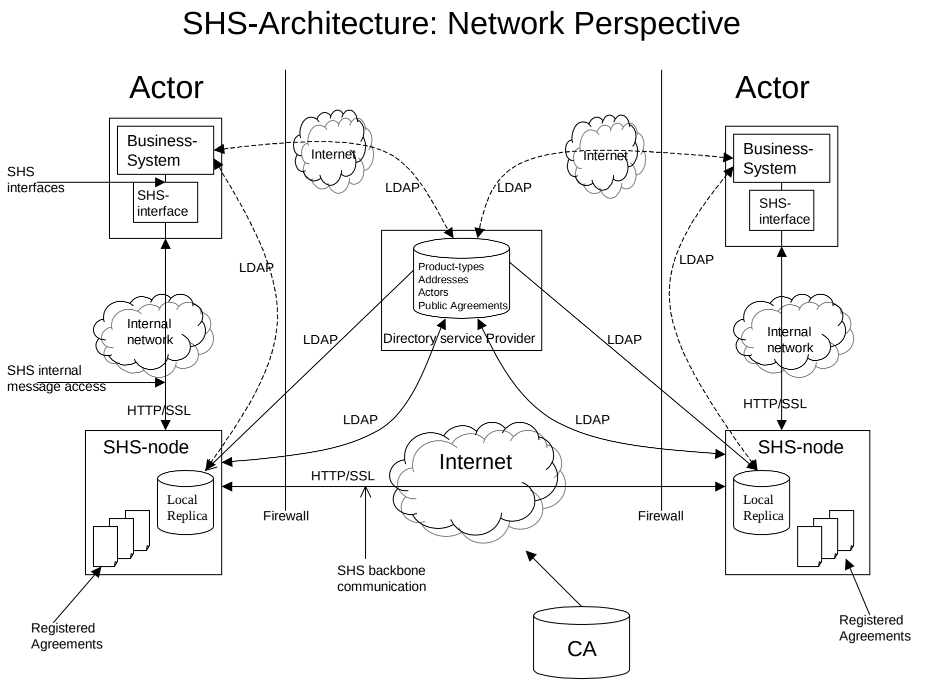
#### *2.5.2.3 Transport layer*

The transport layer defines the HTTP/SSL interaction between two SHS nodes. The security function includes strong authentication of communicating parties and transport encryption using SSL.

The transport layer interface is not defined within SHS.

### 2.5.3 Network perspective

The network perspective describes the relationship between SHS-nodes, clients and different configuration possibilities.



All communication between SHS nodes uses HTTP/SSL. The business application uses one of the SHS interfaces to communicate with the SHS node. The global directory is accessed using LDAP. Normally, a firewall will separate the actor’s internal network and the Internet. HTTP/SSL communication will have to consider the configuration of firewall functionality.

Each SHS node must have a unique identity. It must also have a valid Internet domain name and a current server certificate.

The SHS internal message access defines how messages are retrieved from an SHS-node. The service (on the SHS-node) that is responsible for the retrieval of messages is called the distribution service and is accessed on TCP-port "SHS Internal". See reference [Protocols].

The SHS Backbone Communication defines how SHS-nodes exchange messages. The service (on the SHS-node) that is responsible for this interaction is called the receive service and is accessed on TCP-port "SHS backbone". See reference [Protocols].

#### *2.5.3.1 Node configuration scenarios*

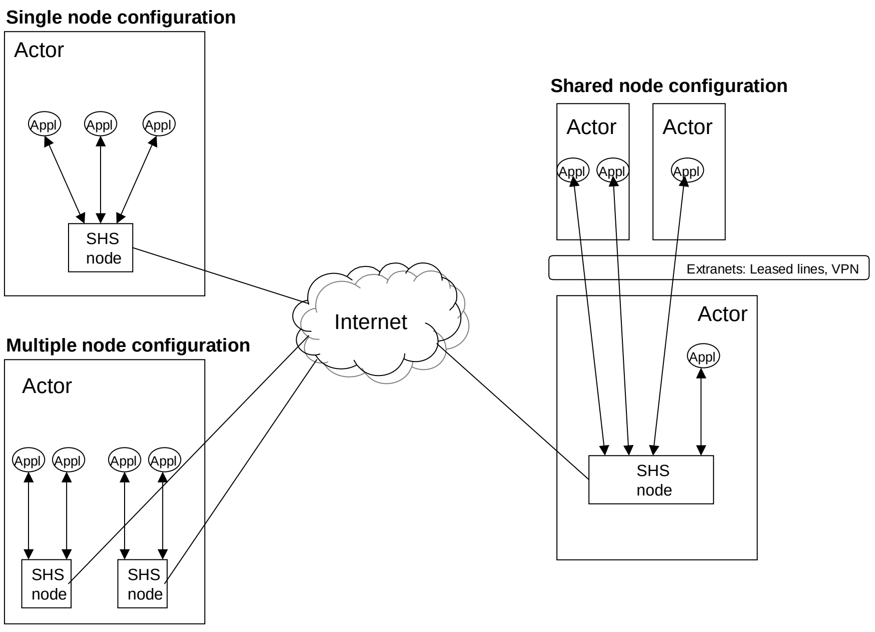
The figure below indicates three different node configuration scenarios.

1. Single node configuration. Normally, an actor has one SHS node connecting the actor's business application.

2. Multiple node configuration is used when an actor configures more than one SHS node. The reason for this is to load balance by directing certain product types to a dedicated SHS node.

3. Shared node configuration is used when an actor (or actors) lets another actor operate the SHS node. In this manner organisations can share the costs of acquiring and operating the SHS MS functionality. The actor owning the SHS-node handles all agreements.

The SHS network defines the collection of SHS-nodes. A single SHS interchange can only involve 2 SHS nodes, that is the SHS architecture does not allow for an SHS node to act as a "transit" node.



## 2.5.4 Security perspective

SHS provides for end-to-end security using encryption and signatures. In an SHS context we define three levels of security:

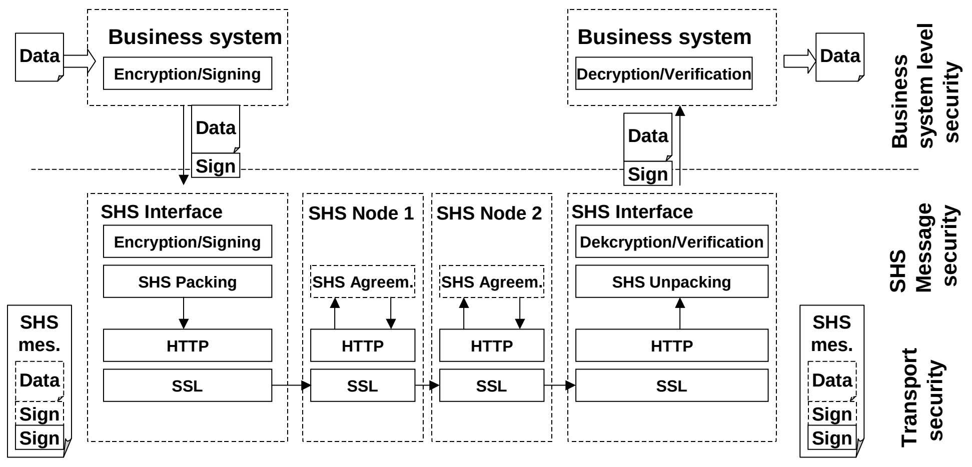
- **Business system level security** defines the security level that the business system choose to impose on the data to be communicated. SHS will not be involved in handling this level of security and will consider business data in the same way whether it is secured or not.
- **SHS transmission security** defines an end-to-end security which means that SHS encrypts and/or signs messages on the sending side and decrypts and/or verifies at the receiving side.
- **Transport security** defines the use of SSL at the HTTP level where each hop on the network between clients and SHS-nodes are secured using SSL.

Any of the three security levels can independently be activated.

The use of SHS security will require a Public key Infrastructure (PKI) with Certification Authority (CA) functionality and administration of certificates.

The figure below indicates the use of security mechanisms within SHS.

Security Perspective



The compulsory use of agreements within SHS is also an aspect of security. This prohibits communication to take place that is not previously agreed on.

Refer to chapter 9 of this document and reference [CA] for a more detailed description of SHS security.

# 3 Interfaces

There are four types of SHS interfaces defined. A detailed description of the interfaces can be found in reference [API].

1. FaP – Format and Protocols. This interface is a description of the low-level protocols and formats used in SHS communication. Wire level protocol is HTTP with SSL/TLS. The message format is based on MIME and S/MIME structures and XML (the label).

   By publishing these ”wire” level protocols it is possible for anyone to construct a program that is able to communicate with an SHS node. See reference [Protocols].

2. External processes (programs). This interface is composed of a couple of programs that can be invoked from applications, shell-scripts etc to communicate with an SHS node. The interface does all packing/unpacking of data and handles security issues etc. See reference [API].

3. API – Application Programmers Interface. An archive of functions callable from the C and Java languages for communication with an SHS node. A tight integration between an SHS-node and business systems are possible with these APIs. The C-API is specified in a pseudo format, hence different vendors implementations might differ. See reference [API].

4. Web Services – An easy to use interface enabling both synchronous and asynchronous transfer. This interface has limitations compared to the C and Java API’s. See reference [API].

An application program can use a combination of the interfaces above.

The interfaces support only simple SHS messages. Support for compound SHS messages are not part of SHS basic services.

For SHS to be able to call an application when performing a synchronous call there needs to be an interface to the application supplying the synchronous response. This is performed by a plug-in to the SHS messaging service. This plug-in will handle the call to the application and return the response to SHS which in turn will be able to provide the synchronous response to the caller. The functionality of plug-ins are not specified by SHS. Nevertheless, it is a prerequisite for synchronous communication and vendors of SHS functionality need to supply this plug-in mechanism. See further section 4.2.

# 4 Communication mode

This section describes in detail the use of asynchronous and synchronous modes of operation. Refer to section 2.5.1 for an overview of services and communication patterns.

## 4.1 Asynchronous operation

In asynchronous mode the sender dispatches an SHS message to SHS and the processing of the message is queued within SHS. The sender only waits till the message is queued which is indicated by a protocol acknowledgement. Successful acknowledgement indicates that the receiving SHS now is responsible for the message.

An asynchronous message can be of four types (called sequence types). These are *event*, *request*, *reply* and *adm*.

- *Event* is used when the sender sends only one message and does not require a reply to this message. All the “One Way” scenarios described in section 2.5.1 will use the sequence type *event*.
- *Request* is used when the sender sends a message that will require a reply paired to this request. The “with response” scenarios in section 2.5.1 will use the sequence type *request*.
- *Reply* is used to indicate that the message is a reply to an earlier request. The “with response” scenarios in section 2.5.1 will use the sequence type *reply* for the response message.
- *Adm* is used for SHS management messages, which are error messages on the SHS protocol level, delivery confirmations and agreement exchanges. These are also called special product types. Refer to section 8.4.1.

Note that although asynchronous mode supports a reply/request scenario it should not be confused with the synchronous mode where the reply is expected within seconds and the sending application waits till the response arrives.

A business system that sends an SHS message can request a delivery confirmation. This is configured in the agreement between the two parties. The acknowledgement is created when the message is fetched by the receiving application and it is sent as an SHS management message (*Adm*).

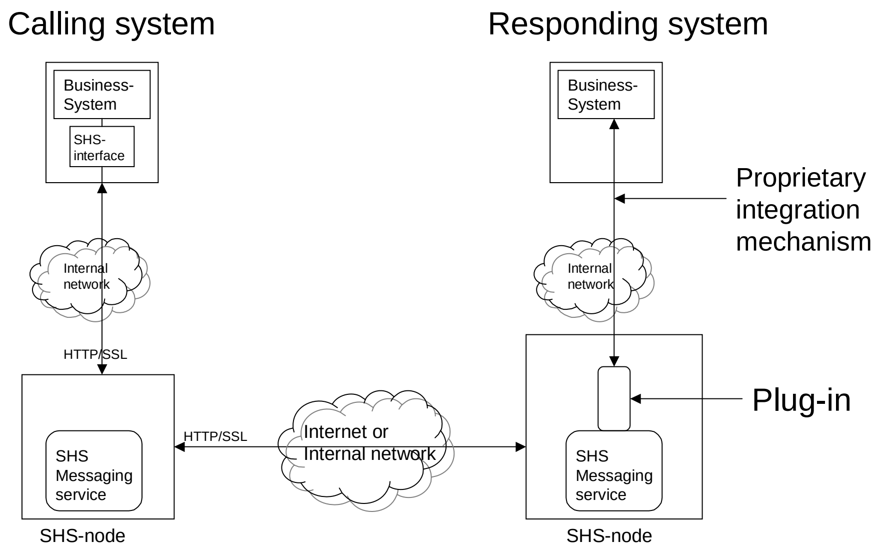
All steps in the asynchronous process are logged within the SHS messaging service and the history list of the message label is updated. This enables SHS messages to be traced through the SHS network.

## 4.2 Synchronous operation

In synchronous handling all processing of the request is done while the sender is waiting, that is a blocked request. A response is normally returned in a very short time, mostly a fraction of a second. In synchronous communication the SHS message is never stored on disk. If the call fails an error will instantly be reported to the caller.

To guarantee good enough synchronous performance, means to avoid the lengthy protocol initialisation sequences should be applied. The SSL/TLS protocols have support for “session reuse”. If session reuse is enabled then a full authentication is done only the first time a connection is established. For the communication that follows a much faster process can be used without sacrificing security. A requirement for this is that session reuse is enabled on both sides.

The synchronous mode of communication needs a way to “alert” and “talk to” the responding application. This is done by a plug-in to SHS. The functionality of a plug-in is not defined by SHS and is thus left to the implementers of SHS messaging services. The figure below describes the role of the plug-in.



The plug-in calls the business system, gathers the response and returns the response back to SHS, which synchronously conveys the response back to the blocked calling system.

# 5 SHS Messaging Service

The SHS Messaging Service is the component that executes the core services of SHS. It is the functionality that resides on the SHS node and is responsible of:

- Receiving messages from applications and other SHS Nodes
- Verifying that messages are valid for SHS transportation
- Delivering messages to applications or other SHS nodes
- Providing a secure transport, that is not loosing messages and not generating duplicates.
- Generating delivery confirmations when requested
- Handling of error conditions by generating error messages, logging errors and placing irresolvable messages in quarantine.
- Routing messages through the SHS network to the receiving application
- Providing trace capabilities using logging and adding trace information to message labels

All communication with SHS messaging service is based on the protocol standards HTTP and SSL/TLS and is described in detail in reference [Protocols].

## 5.1 SHS Messaging Service Operations

The following diagrams outline the SHS messaging service operation. Refer to reference [Protocols] for a protocol perspective of the decisions taken by the SHS messaging service when operating the SHS level protocol.

The diagrams that follow describe the flow through SHS messaging service with respect to:

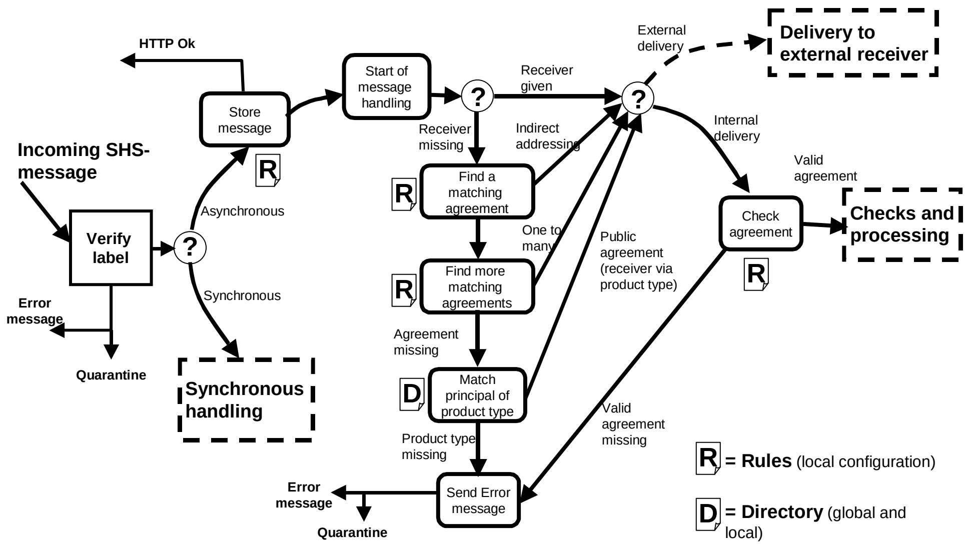
1. An incoming asynchronous message
2. Delivery of an asynchronous message to an external receiver (forward to next SHS-node)
3. Checks and processing done at the receiving SHS-node.
4. Local fetch of message by a local receiver (business system)

5. An incoming synchronous message, including internal communication with responding business application.

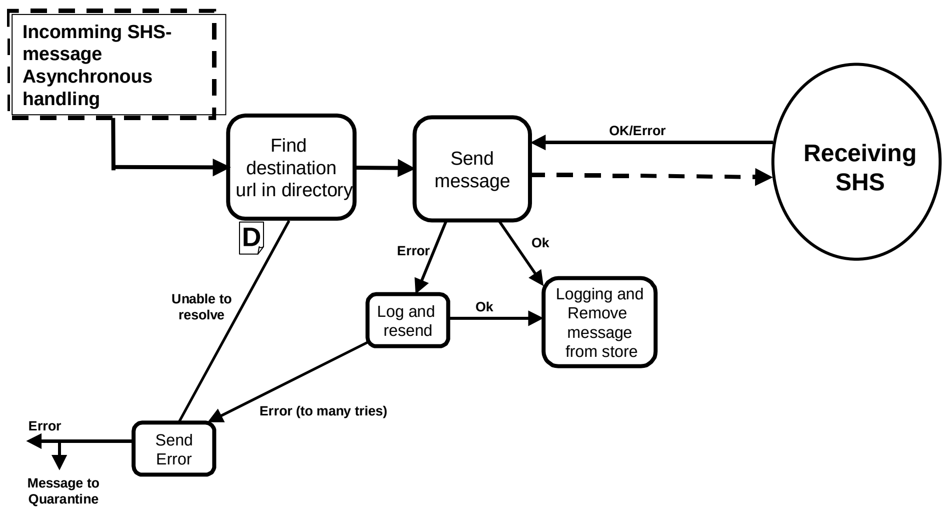
6. Delivery of a synchronous message to an external receiver (call to receiving SHS-node)

### 5.1.1 Asynchronous handling

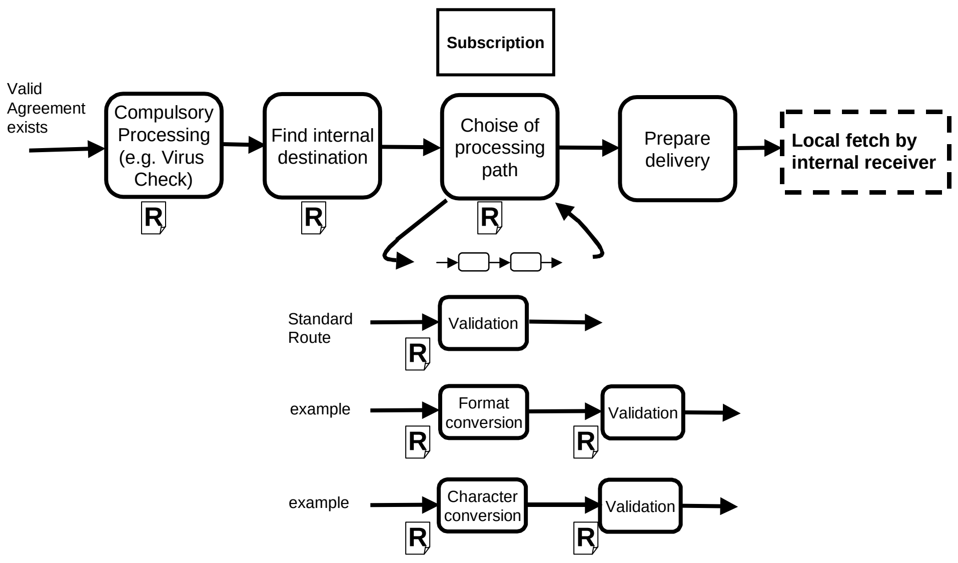
**1. Incoming SHS-message**



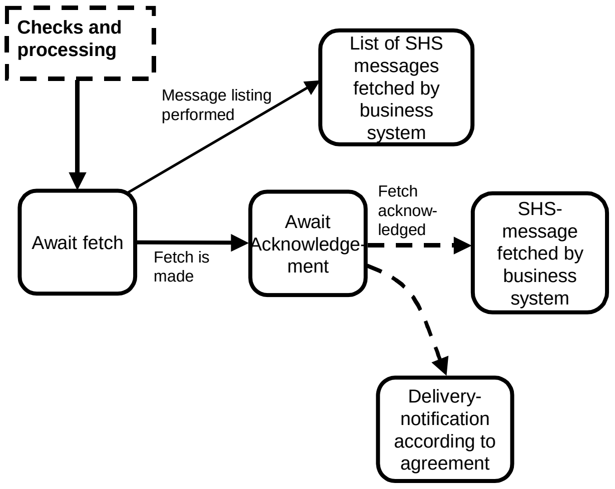
## 2. Delivery to external receiver



## 3. Checks and processing

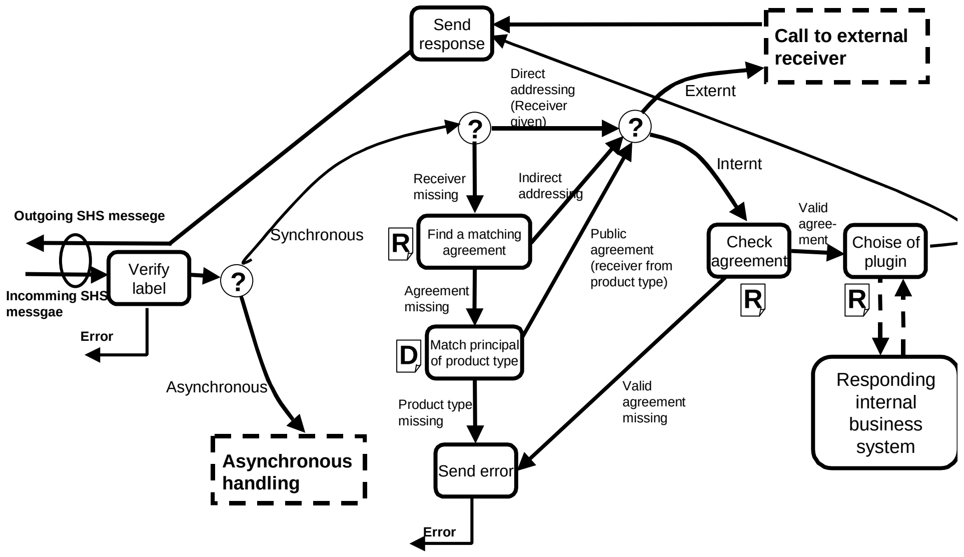


## 4. Local fetch by internal receiver

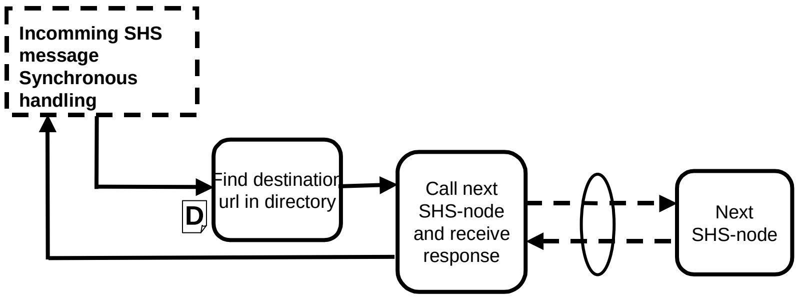

Checks and processing
Message listing performed
List of SHS messages fetched by business system
Await fetch
Fetch is made
Await Acknowledgement
Fetch acknow-ledged
SHS-message fetched by business system
Delivery-notification according to agreement

### 5.1.2 Synchronous handling

## 1 Incomming SHS message



## 2. Call to external receiver


Incomming SHS message Synchronous handling
Find destination url in directory
D
Call next SHS-node and receive response
Next SHS-node

## 5.2 Store and Forward

In asynchronous mode the SHS messaging service operates in a store and forward style. Applications submit SHS-messages to the SHS-node which stores them in a queue until securely submitted to the receiving SHS-node.

In synchronous mode no intermediate queuing is necessary since the communication session is kept open until a reply or a notification is returned.

## 5.3 The role of the agreement

The agreement has a central role for the SHS messaging service. For every transfer through SHS there must be a valid agreement at the receiving SHS messaging service. If no agreement exists or is not configured on the node that receives an SHS-message the transfer is rejected and an error message is returned to the sender. The agreement also plays a role in the routning of SHS-messages. See further routing below.

## 5.4 Routing

Refer to figure 1 and 5 in section 5.1 for routing of asynchronous and synchronous messages.

An SHS node routes a message according to information found in the SHS message label.

The first action when a message arrives to an SHS node is a validation of the SHS message label against its DTD. Every incoming message is also assigned a unique identity as soon as possible. Furthermore, a correlation identity is defined that can be used for match of asynchronous request and replies.

The label carries a status attribute. The status is either test or production. This is used to handle test messages in a production system. The log records whether messages are test or production messages.

If a receiver is stated in the label, SHS uses this address.

If no receiver is found, SHS looks for an agreement with correct product type and actors. If an agreement (or more than one) is found then SHS can get the receiver address from the agreement. If no local agreement is found then SHS searches the global SHS directory for the principal of the product type using a public agreement. SHS then uses the principals' organisation identity to create a valid address in the label.

When the receiver is identified, next step is to locate him. He can be connected to the local SHS or belong to another SHS. Every SHS knows (through a configuration) which organisations it serves, and can thus decide if the receiver is local or remote.

If the intended receiver is locally connected then an agreement for this product type and this receiver is looked up. A check is done that the agreement applies to this sender, otherwise an error message is returned.

In case there are multiple agreements there can be more than one receiver for a message. In this case the message is copied in the required number and each subscriber gets his own copy. Ability to trace the transport is preserved by writing the original messages identity to the label history list. The main receiver gets the original. Forwarding check should be done (on the sender side) if a message lacks receiver address and for all messages received from another SHS.

An incoming message in asynchronous mode is dispatched to a processing path. The product type tag and optionally other information tags control choice of processing path.

## 5.5 Error handling

When SHS detects an error the sender is acknowledged by an SHS management message. The message is stored (placed in quarantine) for manual resolution. If possible, error messages are sent back in direct connection to the call on the HTTP level. Error handling is covered in detail in reference [Protocols].

## 5.6 Logging and trace capabilities

SHS adds the unique message identity to the history list of the label. It is always possible to trace a message through the label history list.

A log is kept with information about every exchanged message. Items in the log are indexed with the message unique identity. Other information in the log include timestamps, sender, receiver, external addresses, message type (synchronous/asynchronous), message size and outcome of any processing activities performed.

## 5.7 Extended services and Plug-ins

When a message is submitted to a processing path, SHS has the ability to call external routines (plug-ins) for processing. See figure 3 in section 5.1. Examples of processing steps can be formatting services or character conversion services. The possibility to call external routines makes SHS extensible and enables the addition of extended services. See chapter 11.

In synchronous communication vendor specific plug-ins are necessary in order to communicate with the responding application. See further section 4.2.

# 6 Addressing

Addresses and product types are constructed according to a pattern described in reference [DTD]. Following this pattern ensures that an address is a globally unique identifier of an SHS node, an internal business system, a service etc.

There are two types of addresses used in SHS communication. These are basic addresses and external addresses.

SHS basic addressing defines the basic addressing between systems in an SHS environment while as external addressing enables external parties to be identified in an SHS message exchange. SHS basic addressing uses the *to* and *from* fields in the SHS message label. SHS external addressing uses the *end-recipient* and the *originator* fields in the SHS message label.

There are also two different modes of addressing used in SHS communication. These are direct addressing and addressing by content type. Both of these modes make use of basic and external addresses when applicable.

The label of an SHS message contains fields (tags) for sender (*from*), receiver (*to*) and product type. An SHS message can be composed of:

- One or more products of the same product type.
- One or more products of different product types. In this case only direct addressing can be used. This is not part of SHS basic services, see section 11 extended services.

## 6.1 Basic addressing

An SHS basic address is composed of two parts, the actor and the internal ID. The internal ID is used to identify a business system internal to an organisation. Using only the actor part makes it possible to hide internal structures from the communicating party. It is **strongly** recommended to use only the actor part between organisations, but using the internal ID is also allowed but **strongly** discouraged. If the internal ID is specified, SHS must use it and not try to find the receiver from using only the actor part. The internal ID will always be resolved at the receiving SHS and the syntax of the internal ID is an issue for local configuration.

When an error message or a delivery notification is sent back to the sender, SHS changes places on the *to* and *from* fields.

## 6.2 External addressing

The purpose of external addressing is to make it easier for external information creators to assign an *originator* identity to the communicated message, or conversely, to point out an external *end-recipient* of the message. The addresses are called external because they are not part of the SHS basic addressing scheme. The content of the external addresses is of no concern for SHS and is up to the application. An example is when an individual fills out a form on the web, the web-server will assign the address of the web-server application as the SHS basic address (*from* field), but use the persons social security number as the external creator address (*originator* field). This originator field will then be conveyed to the receiving application and hence be used as the identity of who actually filled out the form.

The only case when SHS needs to consider the external address fields is when returning a delivery notification or an error message. In this case SHS changes places on the *originator* and *end-recipient* fields in the same manner as it changes places on the *to* and *from* field in basic addressing.

External addressing allows a creator of an SHS message to dispatch the SHS message with only the *originator* field filled out and the *from* field left empty. The creator does this because it has no knowledge of the basic SHS address of the actor it dispatches the message to. In this case the actor must assign the from address of the actor before further conveying the message.

See appendix in chapter 12 for clarifications on how to use basic and external addressing.

## 6.3 Direct addressing

When using direct addressing the sender writes the receivers address in the label. The SHS-message is routed directly to the receiver based on the address in the label. The following table illustrates the use of direct addressing where the receiver address is stated.

| Party | Actions taken | Example |
|---|---|---|
| Sender | The sender states his full address including any internal part.<br><br>States receiver’s global address: “urn:X-shs:9867856345”.<br><br>States message type. | `<from>urn:X-shs:2021000985.fb.1998</from>`<br>`<to>urn:X-shs:9867856345</to>`<br>`<product>(urn) </product>`<br><br>`rsv=2021000985`<br>`rfv=9867856345` |
| SHS 1 | Uses product type and receiver address to find receivers physical address in the global SHS directory. Sends the message to the receiving | `Forwards message.` |

| | | |
|---|---|---|
| | SHS node. | |
| SHS 2 | Dispatches message to processing path according to label information.<br><br>Possible distribution to more than one business system.<br><br>Internal business systems that could receive an SHS message are listed in a local table. | `<from>urn:X-shs:2021000985.fb.1998 </from> <to>urn:X-shs:9867856345.kund.99 </to> <product>(uuid)</product>` |
| Receiver | The receiver gets his message. | `Get the message and do whatever is the responsibility of the business system.` |

The “One to Many” communication scenario is a special case of direct addressing. In this case the sending business system leaves the receiver field empty, where after the adjacent SHS node copies the SHS message and fills in the direct addresses based on the agreements defining the “One to Many” scenario.

## 6.4 Content based addressing

When addressing by content is used, the sender marks the message with product type but do not state any receiver.

In a “One to Many” communication scenario agreements for this product type must exist and be defined within the SHS messaging service. One agreement for each receiver.

In a “Many to One” scenario there must exist a public agreement in the global SHS directory.

What is common is that an agreement must exist in order to match the product type and receiver/s.

The following table illustrates the use of content based addressing where the product type is stated and possibly more than one receiver is present.

| **Party** | **Actions taken** | **Example** |
|---|---|---|
| Sender | The sender states his full address including any internal part.<br><br>The sender states no receiver. An | `<from>urn:X-shs:2021000985.fb.1998</from> <product>(urn) </product>`<br><br>`rsv=2021000985 rfv=9867856345` |

| | | |
|---|---|---|
| | agreement exists for the product type, which means that content based addressing is used.<br><br>States product type of the message. Note that the message can be composed of more than one product, all of the same product type. | |
| SHS 1 | Uses product type and address. No receiver indicates that addressing by content type should be used.<br><br>Uses agreements to find receiver(s) for this product type.<br><br>For each receiver, finds his physical address (URL) in the global SHS directory and sends the message.<br><br>Note that SHS-1 can send the same product to more than one receiver according to the “One to Many” scenario (this is not showed in the example). | `“fb-avisering” to “rfv”`<br><br>`<from>urn:X-shs:2021000985.fb.1998</from>`<br>`<to>urn:X-shs:9867856345</to>`<br>`<product>(urn)</product>` |
| SHS 2 | Dispatches message to a processing path according to label information.<br><br>Possible distribution to more than one business system.<br><br>Use agreement to find receiver(s) for this product type. | `SHS at rfv handles`<br><br>`<from>urn:X-shs:2021000985.fb.1998</from>`<br>`<to>urn:X-shs:9867856345.kund.99 </to>`<br>`<product>(urn)</product>` |
| Receiver | The receiver gets his message. | `Get the message and do whatever is the responsibility of the business system.` |

## 6.5 Rules for addressing combinations

When the sending application specifies the receiver addresses the following rules apply:

| **Label field “to”** | **Label field “End Recipient”** | **Product type(s) Single or Many** | **Addressing scheme used** |
|---|---|---|---|
| Empty | Filled in or Empty | Single | By content |
| Filled in | Filled in or Empty | Single or Many | Direct |

| | | | |
|---|---|---|---|
| Empty | Filled in or Empty | Many | Not allowed |

Note: Use of many product types in one SHS message implies the use of a compound SHS message. This is not part of SHS basic services.

When the sending application specifies the sender addresses the following rules apply:

| **Label field "Originator"** | **Label field "From"** | **Addressing type used** |
|---|---|---|
| Filled in | Filled in | External |
| Filled in | Empty (sending SHS-actor fills in this field) | External |
| Empty | Filled In | Basic |
| Empty | Empty | Not allowed |

# 7 Message structure

The data structure transported by the SHS network is named an SHS message. An SHS message is composed of a label coded in XML and one or more data parts (the payload). The label contains control information for the transport system. It is described and documented in a DTD (Document Type Definition). As an envelope for the different parts of an SHS message S/MIME is used. All or part of the payload of business information can be digitally signed and/or encrypted. Transported information is declared “products” and a ”product type” defines each product. An example of a product type is: “Income Tax Form”.

## 7.1 The SHS message

The structure used for transporting information is called an SHS message. An SHS message can contain one or more products.

Structure of an SHS message is defined by the following (BNF) production:

```
SHS-document      ::=     SHS-simple | SHS-compound
SHS-simple        ::=     SHS-label (Data)*
SHS-compound      ::=     SHS-label (SHS-document)*
```

Note that the BNF production uses the name “SHS-document” with the meaning SHS message. The historical explanation to this is a name change from document to message.

An SHS message is either a simple or a compound type of message. Only the simple SHS message is part of the SHS basic services. A simple message starts with a label and one or more data parts follow the label. A compound message also starts with a label that is followed by a hierarchical structure of embedded SHS messages. The compound message handling is part of the SHS extended services.

The label structure is always the same irrespective of the message content.

Data can be anything. It is recommended to use MIME types and subtypes conformant to the IANA register but use of other types can also be used. The default format is text XML

If we need to sign and/or encrypt a data part in addition to the transport security of SSL, S/MIME is used. This is called message security. Note that the label never is signed or encrypted.

### 7.1.1 The label

The label contains control information for the SHS-message. It includes:

- Addressing information

- Associated product type packaged in the message
- Informational field such as content description
- Version, timestamp and history list
- Unique identification of message and correlation id for response
- Name of agreement that controls the transfer
- Transfer mode, that is synchronous or asynchronous
- Message type, simple or compound
- The sequence type, that is event, request, reply or administrative message
- Status field that indicates test or production

### 7.1.2 The simple SHS message

This structure is the common one. It could be used for both asynchronous and synchronous communication modes. The simple SHS message can contain only one product type.

The data part(s) are separated from the label in a multipart/mixed MIME structure. The sender side SHS API creates the MIME structure and vice-versa when a business system receives the message. See reference [API].

If a data part is signed the signature is stored after the data part according to the S/MIME structure.

### 7.1.3 The compound SHS message

A compound SHS message is an SHS message containing a hierarchical structure of embedded SHS messages. Functionality for packing and unpacking of these structures is currently not part of the SHS interfaces.

Compound messages can be handled by a component outside the SHS messaging service. When a SHS node receives a compound message it can delegate the unpacking process to the add-on module for compound message unpacking and vice-versa for packing.

The compound structure could only be used in asynchronous communication. This function is defined as an SHS extended service.

# 8 Information entities

This section defines the role of the actor, product types and agreements and relations between these definitions. A separation of what is globally defined (in the common SHS-directory) and what is a matter of local configuration is also made.

## 8.1 Common/global information

Common information should be used in an equivalent manner by all SHS implementations. It has been a goal in the design of SHS to minimize the use of common information. Information defined as common/global includes:

- Information about actors.
- Addresses
- Information about product types and pointer to detailed product type descriptions.
- Public agreements, which define a many to one communication scenario.
- Security information related to CA and certificate directories (are described in another part of the message). Security related information is not stored in the SHS directory.

## 8.2 Locally defined information

Information that needs to be configured locally on the SHS-node is:

- Agreements that are not public and thus define a bilateral exchange.
- Product types to be exchanged in a bilateral agreement. These product types can also be registered in the SHS directory.
- Locally defined addresses.
- Subscriptions, which is a local option to copy the reception of an SHS-message to other receivers.

## 8.3 The Actor

The SHS Actor is a part exchanging SHS messages with other actors. The actor has an SHS node (or accessing an SHS-node as a service) and has connected one or more of its business applications to the SHS network.

Information about the SHS actor must be stored in a global SHS directory. The SHS actor can be an authority or any type of organisation. All participating organisations must be registered in the global SHS directory. The directory contains information about the actor such as description, postal address, telephone number etc. Each actor is responsible for it’s own information and all product types it owns (is principal for).

For governments, their organisation identity (organisation number) is used as unique key in the SHS directory and also in agreements.

## 8.4 Product type

The SHS Product type is a well defined structures of the artefacts transported in the SHS network, mainly various business documents. There is always an SHS actor who is the owner (the principal) of a product type.

Information about the product type is stored in the SHS directory. Examples of product type information in the directory are name, UUID, description and principal. There is also a pointer to an XML description file of the product type.

A product type is made up of one or more SHS data parts that together constitute a business product, e.g. an “Income Tax Form” or a “Report of Illness”.

The product type owner (principal) is responsible for it’s definition, publication and update. The principal should store base information about his product types locally and publish them in the global SHS directory. All product types used in public agreements (see below) must be published in an SHS directory. The SHS directory contains information about the principal and it is used to find the principal’s address when addressing by content type is used.

Product types have a unique identity in a UUID format, see further reference  [DTD].

A product is an instantiation of a product type and contains the actual business data to communicate. It can be a temporal memory structure in a business system that can be mapped to some data storage structure. A product can also be a request/reply pair. The request can be an online message, the reply an online answer or one or more asynchronous messages.

### 8.4.1 Special products

Three message types are defined as special products. These are error messages, delivery notifications and agreement exchanges. These message types are exchanged without agreements between actors. See reference [DTD] for DTD definitions.

Special products can be seen as being part of the SHS protocol and differ largely from the products that define the business interaction between actors.

## 8.5 Agreement

The SHS Agreement is a set of rules that governs the handling of a given product type or product types between two actors. An agreement declares for example transfer cost, confirmation requirements and if a product is composed of a reply/request scenario. The principal (one of the actors) is the owner of the agreement. The principal of an agreement is responsible for storing and distributing it to other parties involved. No communication between SHS actors is possible without an agreement.

Agreements used in SHS are not agreements from a legal point of view. However, in many cases there will be a legal document relating to the SHS agreement. The SHS agreement should be accessible (in machine readable XML format) to the parties. SHS agreements are an important mechanism in the SHS network.

All addressing in the SHS network are based on agreements. There are two types of agreements. These are public agreements and bilateral agreements. Bilateral agreements are not published in the global directory while as the public agreements are.

### 8.5.1 Public agreement

Open-ended agreements are called “public agreements”. They will be used for communication between an organisation and many other parts, e.g. the Tax Board and all people sending a declaration form. Public agreements enable the “Many to One” communication scenario.

When a public agreement is used one only sending (initiating) actors address is used.

### 8.5.2 Bilateral agreement

The bilateral agreement is set up between a principal and a customer and relate to one or more product types. The parties could be government – government, government – enterprise or other combinations of organisations that communicate by SHS. The communication mode, synchronous or asynchronous, is stated in the agreement.

Note that also communication between two (or more) business systems local to one SHS must use agreements.

### 8.5.3 Exchange of agreements

An agreement is exchanged between the parties as an XML document that is conformant to a DTD defined in reference [DTD]. An exchange of agreement is a special product type (see section 8.4.1).

The original or master copy of an SHS agreement is registered and stored by the principal. An agreement must have a unique identity.

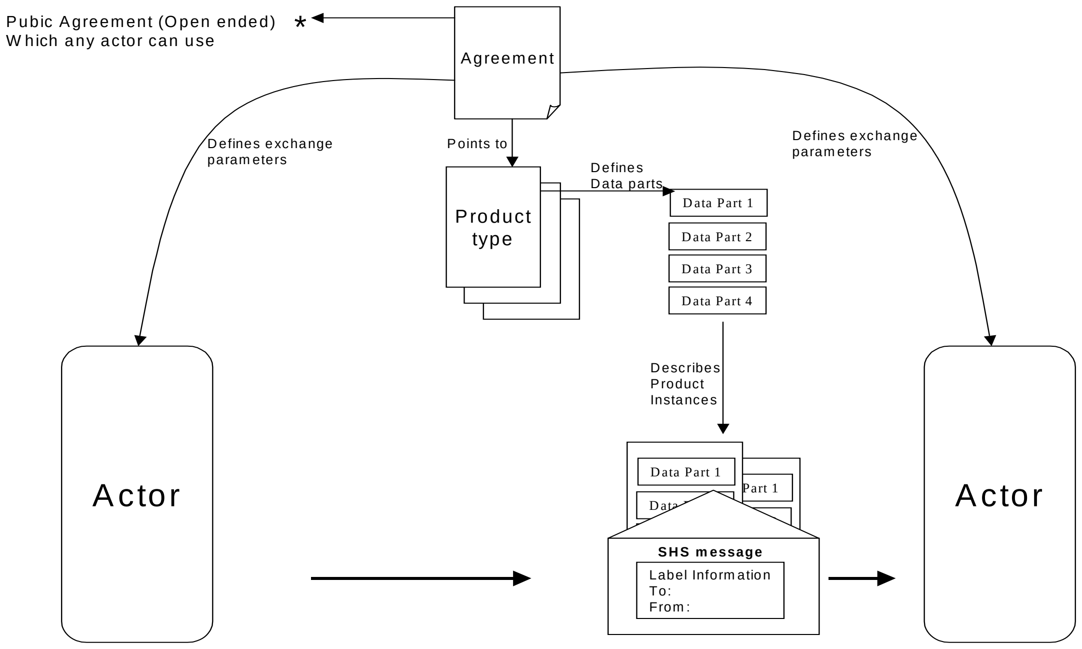
An agreement is updated when the principal creates a new copy with a new unique identity. The new agreement is distributed to the other party. Both parties have a copy of the agreement.

## 8.6 Information entity relationships

The figure below illustrates the relationship between the actor, the product type, the agreement and the actual SHS message.

The agreement states the exchange parameters between two actors. In the case of a public agreement any actor can initiate a transfer based on this open-ended agreement.

The agreement relates to one or more product types. The product type defines the SHS data parts to be communicated using that particular product type. One or more product instances are packaged in an SHS message for transfer. The SHS message label carries the necessary information to route the message to the destination.


Pubic Agreement (Open ended)
Which any actor can use
*
Agreement
Defines exchange parameters
Points to
Defines exchange parameters
Product type
Defines Data parts
Data Part 1
Data Part 2
Data Part 3
Data Part 4
Describes Product Instances
Actor
Data Part 1
Part 1
SHS message
Label Information
To:
From:
Actor

# 9 Security

As described in the security perspective in section 2.5.4 SHS defines three levels of security. These are:

- **Business system level security** defines the security level that the business system choose to impose on the data to be communicated. SHS will not be involved in handling this level of security and will consider business data in the same way whether it is secured or not.

- **SHS message security** defines an end-to-end security which means that SHS encrypts and/or signs messages on the sending side and decrypts and/or verifies at the receiving side.

- **Transport security** defines the use of SSL at the HTTP level where each hop on the network between clients and SHS-nodes are secured using SSL.

SHS is responsible to accommodate for SHS transmission security and transport security. These are described in additional detail below. Also refer to reference [CA] for detailed information on the use of certificates within SHS and the requirements on a certification authority.

The use of agreement is also an aspect of security. It limits the communication to take place only between actors that have previously worked out an agreement and configured these agreements within SHS.

## 9.1 Message security

End-to-end encryption and signing of one or more data parts in a message are done at the client. This signing and encryption of data parts are independent of the transport encryption. Certificates (public keys) are enclosed with signed data parts, as stated in S/MIME.

Indication that parts of a message are signed and/or encrypted can only be concluded from the S/MIME structure, there is no mark-up in the label for this.

However, signing and/or encryption of data parts are controlled from the product type definition. These functions are defined at the data part level, not the product type level, i.e. a product can be composed of more than one part and security requirements can differ from part to part.

The principal prescribes which security level(s) should be used for a message.

## 9.2 Transport security with SSL

SSL can be used for all communication between SHS components. SSL based communication should use mutual authentication with X.509 certificates. Both encryption and strong (mutual) authentication with SSL are used. Used asymmetric keys are at least 1024 bits, symmetric keys at least 128 bits.

Certificate and their keys could be soft (stored on disk) or hard (stored in a smart card).

Both parts must check that the other parts certificate is valid, that it is issued by a trusted Certification Authority (CA) and that the certificate is not on a revocation list.

## 9.3 The CA service

A CA (Certification Authority) is responsible for:

- Create the private and public keys (unless they are created by different users/systems).
- Create soft and hard (smart cards) certificates and publish them in a public CA directory.
- Publish lists of revoked certificates in a form that can be replicated to the SHS nodes.

# 10 Directory

The SHS Directory is a common repository where all SHS enabled applications and actors may find information about organisations, services (products) and addresses. It is assumed that the directory is implemented using LDAP version 3 (Lightweight Directory Access Protocol) and that specific tree structure and object classes (types of information) is handled by the directory as well as clients.

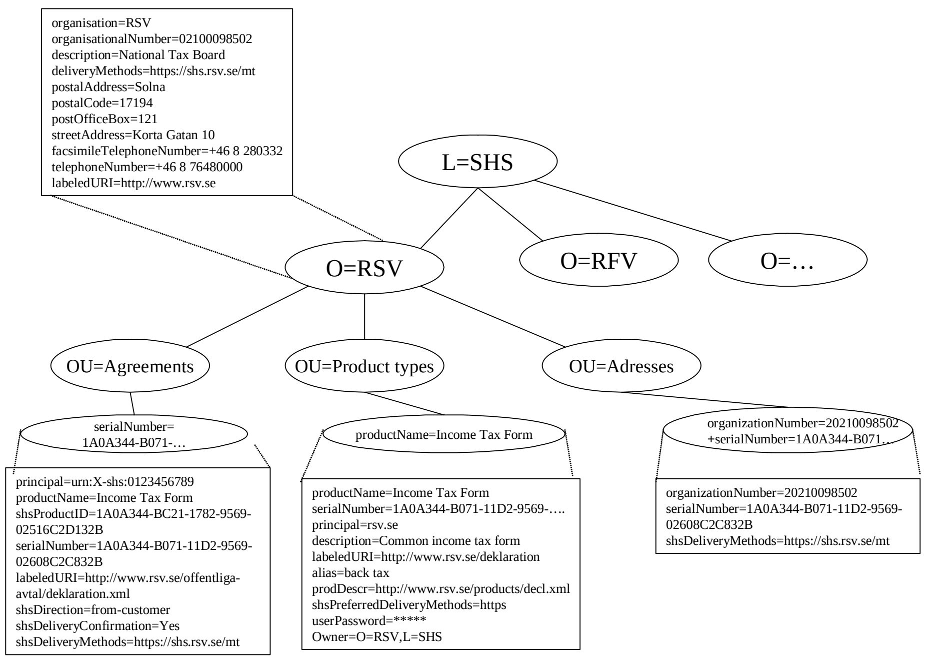
The SHS enabled systems must be able to recognize more than one directory service provider. This will allow organisations connected to SHS to publish directory information in one directory only.

An SHS node can choose to store a local replica of the SHS directory to obtain satisfactory performance.

A detailed specification of the SHS directory can be found in reference [DIR].

## 10.1 Directory structure

Sub tree structures are used to provide separated management domains, i.e. each authority should be allowed to manage it’s own entities in the directory. The tree structure of the directory is shown in the figure below.



organisation=RSV
organisationalNumber=02100098502
description=National Tax Board
deliveryMethods=https://shs.rsv.se/mt
postalAddress=Solna
postalCode=17194
postOfficeBox=121
streetAddress=Korta Gatan 10
facsimileTelephoneNumber=+46 8 280332
telephoneNumber=+46 8 76480000
labeledURI=http://www.rsv.se

principal=urn:X-shs:0123456789
productName=Income Tax Form
shsProductID=1A0A344-BC21-1782-9569-02516C2D132B
serialNumber=1A0A344-B071-11D2-9569-02608C2C832B
labeledURI=http://www.rsv.se/offentliga-avtal/deklaration.xml
shsDirection=from-customer
shsDeliveryConfirmation=Yes
shsDeliveryMethods=https://shs.rsv.se/mt

productName=Income Tax Form
serialNumber=1A0A344-B071-11D2-9569-….
principal=rsv.se
description=Common income tax form
labeledURI=http://www.rsv.se/deklaration
alias=back tax
prodDescr=http://www.rsv.se/products/decl.xml
shsPreferredDeliveryMethods=https
userPassword=*****
Owner=O=RSV,L=SHS

organizationNumber=20210098502
serialNumber=1A0A344-B071-11D2-9569-02608C2C832B
shsDeliveryMethods=https://shs.rsv.se/mt

## 10.2 Objects and attributes in the SHS directory

Information about actors, products, addresses and public agreements are stored as objects in the directory. These are defined by their respective object classes:

1. Actors are defined by the object class organization. The attributes of this class include information on actors name, description, postal address information, phone numbers, web address and organisation number.

2. Product types are defined by the object class shsProduct. The attributes of this class include information on name of product type, identity (UUID), principal, description of product, pointer to more information, searchable keywords, pointer to XML description, delivery method, password for directory updates and owner of the product.

3. Addresses are defined by the object class shsAddresses. The attributes of this class include information on how to find the address of a service based on the organisation number and a product type UUID.

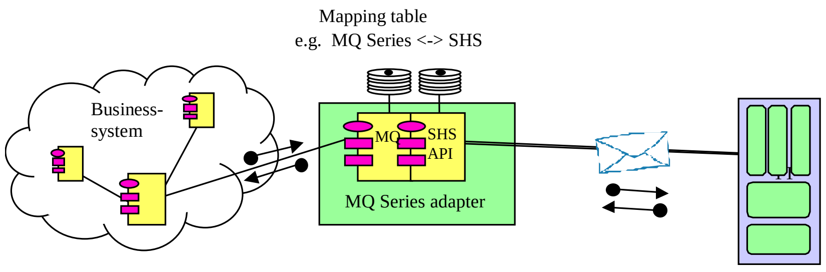
4. Public agreements are defined by the object class shsAgreement. The attributes of this class include information on the principal, product referred by the agreement, pointer to the agreement in XML format, change date of the agreement, communication direction defined by the agreement, description, valid dates and address to SHS service for referenced product.

# 11 Extended services

This section describes services outside the scope of SHS basic services. SHS extended services are not specified in detail and detailed functionality is left to the implementers.

## 11.1 Additional interfaces

Additional interfaces are adapters between SHS and various types of middleware. With these interfaces software developer can use a well-known environment and well known tools while implementing applications that use SHS. For example he can use an IBM MQ Series API and yet the underlying transport mechanism is SHS.



There are of course many products with different API’s, formats and protocols needed by a customer. Adapters for the following products/standards are of interest:

- Web Services
- JMS (Java Messaging Services)
- IBM MQ Series
- BEA Tuxedo/FML
- Microsoft DCOM/COM+
- Microsoft Message Queue
- OMG CORBA & IIOP

This list is not in any priority order, market demand will state the order of implementation.

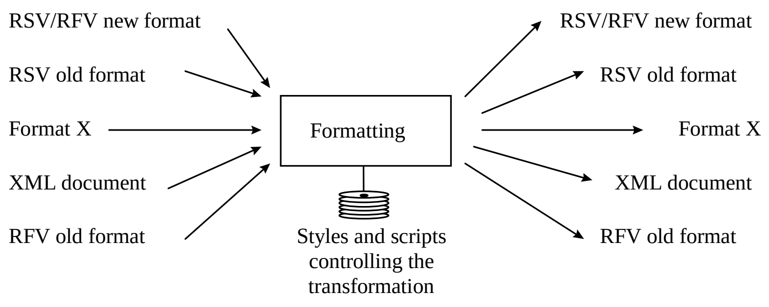
There is no requirement that all adapters support the full functionality in either SHS or the other middleware but any supported subset must be well documented.

## 11.2 Character conversions

The function for character conversion performs a translation between different character sets. Handling of ISO 8859-1, CP 850, EBCDIC CP 00278 is required. Translation tables for other character sets may be defined locally. Support for Unicode ISO 10646 (UTF 8 and UTF 16) is desired.

## 11.3 Formatting

This component performs transformation and formatting between different document and file structures. Format descriptions are stored in machine-readable form in a repository. Automatic transformation can only be done when the semantic content is the same in input and output formats (down translation or one-to-one translation). Format descriptions are in EBNF, XML(Schema) or DTD form. Conversion between older formats (Cobol fix length records, tagged formats other than XML etc) and newer (XML) should be possible to facilitate migration to SHS. Format X in the figure below can be Tuxedo FML or other proprietary formats.



The formatting component should be available for applications also outside SHS as a library component.

The number of possible transformations is very large. Examples are:

- XML
- UN/EDIFACT
- Files with comma as separation (CSV) or similar with other separation characters.
- DIF – Data Interchange Format
- Fixed record length files (Cobol format)

- Tagged file, e.g. "*#tagname data*"

It should be possible to transform all formats in this list to and from each other without losing information. Tools for transformation between arbitrary formats is desirable.

This list is not in any priority order. Market demand will decide the order of implementation.

## 11.4 Compound SHS message

The compound SHS message is not part of SHS basic services. See section 7.1.3 for a definition.

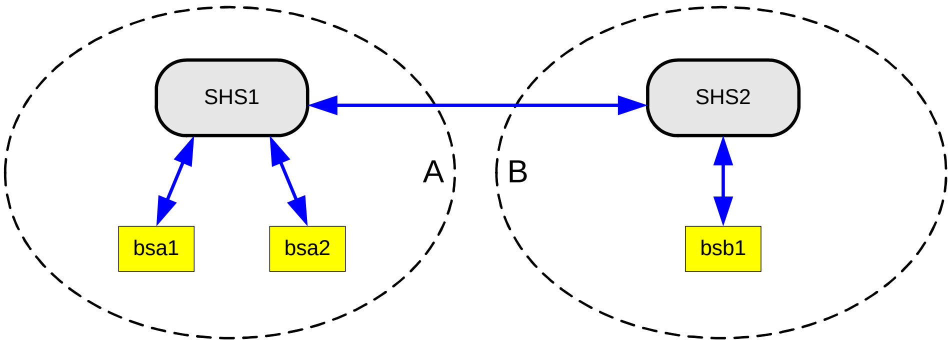
## 12 Appendix: Addressing scenarios

This appendix contains a number of examples of addressing scenarios to clarify the use of basic and external addressing. The description is focused on the addressing of SHS-messages and how SHS handles these addresses

The first section describes addressing between business systems using basic addressing. The second section describes addressing when external parties are involved. A few examples of services (web-interface, client program and mailbox) are used to clarify the use of addressing. Note that these services are not part of the SHS specification and must be seen in the SHS perspective as a business application interfacing SHS and making use of SHS functionality.

### 12.1 Basic addressing (between SHS-connected business systems)

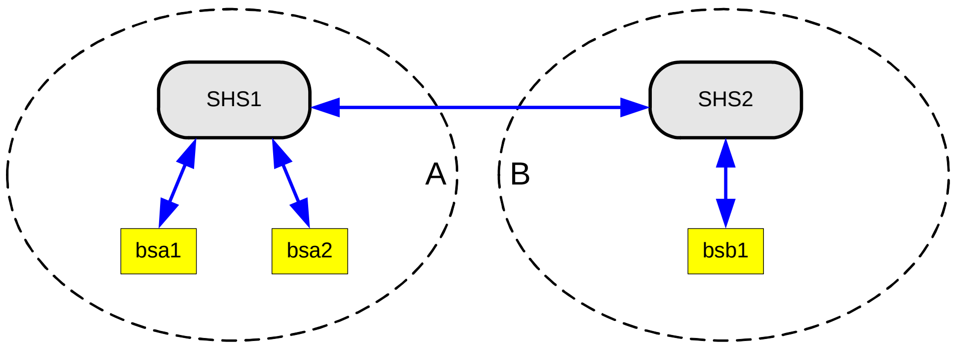
#### 12.1.1 Addressing of a delivery notification to a business system



This scenario describes the handling of delivery notification. Two applications at the same actor send a message to an application at another actor requesting a delivery notification

| | |
|---|---|
| **Parties** | **A** (2021000123), actor (organisation, authority)<br>**B** (2021000124), actor (organisation, authority)<br>**bsa1** and **bsa2**, business systems at **A**<br>**vsb1**, business system at **B** |
| **SHS** | **SHS1**, belongs to **A**<br>**SHS2**, belongs to **B** |
| **Product types** | **P**, a product type published by **B** |
| **Procedure** | Business system **bsa1** sends an SHS-message to **SHS1**. The message has **P** as product type, *urn:X-shs:2021000124* as receiver and *urn:X-shs:2021000123.bsa1* as sender.<br>In the same manner **bsa2** sends a message to **SHS1** with *urn:X-shs:2021000124* as receiver and *urn:X-shs:2021000123.bsa2* as sender.<br>**SHS1** sends these SHS-messages further to **SHS2**.<br>In **SHS2** the receiver address is in both cases rewritten to *urn:X-shs:2021000124.vsb1*. Business system **vsb1** can eventually fetch theses messages.<br>Delivery confirmation is sent automatically by **SHS2** when **vsb1** has fetched a message. The delivery notifications are addresses by changing places on the sender and receiver addresses in the messages. The delivery notifications of the messages sent by **bsa1** ultimately get sender address *urn:X-shs:2021000124.vsb1* and receiver *urn:X-shs:2021000123.bsa1* as soon as it is sent from **SHS2**. The same will apply to **bsa2**.<br>Business systems **bsa1** and **bsa2** can thereafter at **SHS1** list and fetch their delivery notifications. |

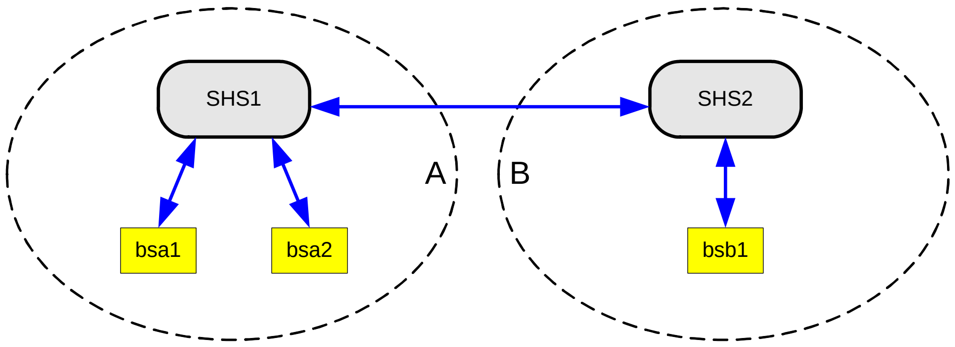
### 12.1.2 Addressing of an error message to a business system



This scenario describes the handling of error messages during transport. It resembles the handling of delivery notification, but the difference is that an error message might need to be sent earlier in the transport change than the delivery notification. This means that the system sending the error message might not have the same amount of information as to where the erroneous message was meant to be sent.

| | |
|---|---|
| **Parties** | **A** (2021000123), actor (organisation, authority)<br>**B** (2021000124), actor (organisation, authority)<br>**bsa1** and **bsa2**, business systems at **A**<br>**vsb1**, business system at **B** |
| **SHS** | **SHS1**, belongs to **A**<br>**SHS2**, belongs to **B** |
| **Product types** | **P**, a product type published by **B** |
| **Procedure** | Business system **bsa1** sends a message to **SHS1** with an erroneous receiver address. In the same manner **bsa2** sends a message to **SHS1** with an erroneous receiver address. The messages both have *urn:X-shs:2021000189* as receiver and *urn:X-shs:2021000123.bsa1* and *urn:X-shs:2021000123.bsa2* respectively as sender addresses.<br>Due to an erroneous receiver address, an error is discovered in **SHS1** and error messages are automatically sent by **SHS1**.<br>The error messages are addressed by changing places on receiver and sender addresses in the messages to be reported erroneous. The error message pertaining to the message sent by **bsa1** ultimately gets as sender the erroneous address and as receiver *urn:X-shs:2021000123.bsa1*. The same will apply to **bsa2**.<br>The business systems **bsa1** and **bsa2** can thereafter at **SHS1** list and fetch their error messages. |

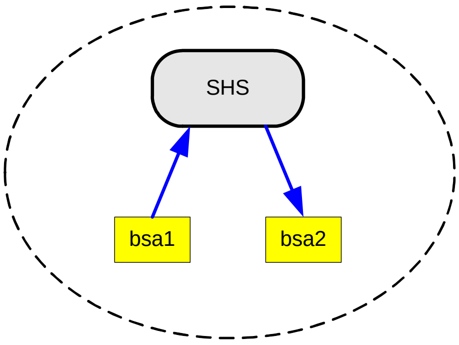
### 12.1.3 Addressing of asynchronous replies to a business system



This scenario describes the handling of a reply/request scenario using asynchronous transfer mode. Two business systems at the same actor send messages with reply requests to another business system at another actor and hence want the replies delivered to the pertinent system. It resembles scenario 1.1.1, with the important distinction that in this case it is the business system that sends the response and not SHS as in the case with delivery notifications.

| | |
|---|---|
| **Parties** | **A** (2021000123), actor (organisation, authority)<br>**B** (2021000124), actor (organisation, authority)<br>**bsa1** and **bsa2**, business systems at **A**<br>**vsb1**, business system at **B** |
| **SHS** | **SHS1**, belongs to **A**<br>**SHS2**, belongs to **B** |
| **Product types** | **P**, a product type published by **B** |
| **Procedure** | Business system **bsa1** sends a message to **SHS1**. The message is sent with **P** as product type, *urn:X-shs:2021000124* as receiver and *urn:X-shs:2021000123.bsa1* as sender.<br>In the same way **bsa2** sends a message to **SHS1** with *urn:X-shs:2021000124* as receiver and *urn:X-shs:2021000123.bsa2* as sender.<br>**SHS1** sends these further to **SHS2**.<br>In **SHS2** the receiver address will in both cases be rewritten to *urn:X-shs:2021000124.vsb1*. Business system **vsb1** can then fetch these messages.<br>When **vsb1** has computed the requests it leaves response messages to **SHS2**. The responses are addressed with *urn:X-shs.2021000124.vsb1* as sender and *urn:X-shs:2021000123.bsa1* and *urn:X-shs:2021000123.bsa2* respectively as receiver addresses.<br>**SHS2** sends these response messages further to **SHS1**.<br>The business systems **bsa1** and **bsa2** can thereafter at **SHS1** list and fetch their response messages. |

### 12.1.4 Internal information exchange within an authority



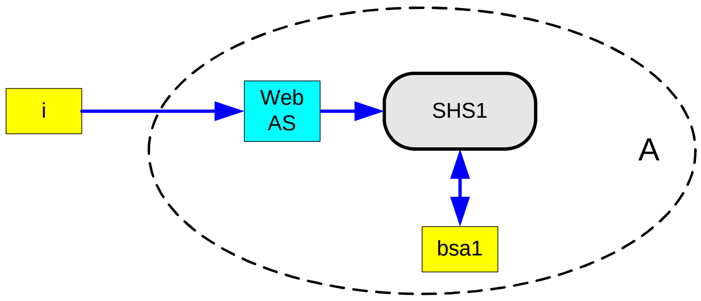
In this scenario a business system will send a message to another business system within the same authority

| | |
|---|---|
| **Parties** | **A** (2021000123), actor (organisation, authority)<br>**bsa1** and **bsa2**, business system at **A** |
| **SHS** | **SHS1**, belongs to **A** |
| **Product types** | **P**, a product type published by **A** |
| **Procedure** | Business system **bsa1** sends a message to **SHS1**. The message is sent with **P** as product type, *urn:X-shs:2021000123.bsa2* as receiver *urn:X-shs:2021000123.bsa1* as sender.<br>**SHS1** sees that **bsa2** is an internal receiver and prepares the message for fetching.<br>Business system **bsa2** can thereafter at **SHS1** list and fetch the message from **bsa1**. |

## 12.2 Addressing for external parties

This section describes a few scenarios where addressing of individuals and small companies is a concern.

### 12.2.1 Dispatching information via a web-interface

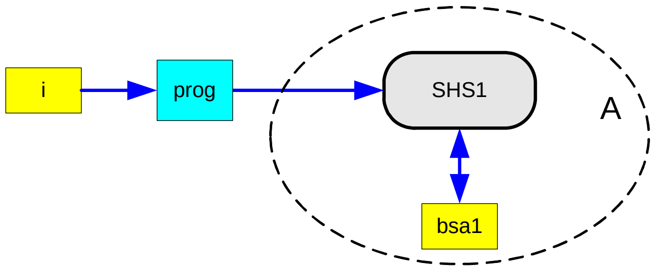


When an authority offers the service to electronically receive information from an individual and small companies/organisations (that are not SHS-actors), an adaptation server is recommended. The adaptation server will supply the services to the individual who will not be concerned with the actual technology. An example of an adaptation server can be a Web-server.

In this scenario an authority has published a product type and established a service on the web where the individual can fill out a form that conforms to the actual product type. The content of this form is then sent as an SHS-message to a receiving application

| | |
|---|---|
| **Parties** | **i** (7209191234), an individual<br>**A** (2021000123), an actor<br>**bsa1**, business system at **A** |
| **SHS** | **SHS1**, belongs to **A**. |
| **Product types** | **P**, a product type published by **A** |
| **Procedure** | Individual **i** connects to the web service and receives an empty form. The individual **i** fills out the form with data pertaining to product type P and also supplies his/her social security number. Sending the form to the web server ends the session.<br>The web server builds an SHS-message and sets *urn:X-shs:2021000123.webas* as sender and *urn:X-shs:2021000123* as receiver. The webservice will also set *pno:7209191234* as originator and **P** as product type. The message is sent to **SHS1** by the web-server.<br>In **SHS1** the receiver address will be rewritten to *urn:X-shs:2021000123.bsa1*.<br>The business system **bsa1** can thereafter fetch the message and via the originator see that individual **i** was the one leaving information regarding product type **P**. |

## 12.2.2 Leaving information using a client program



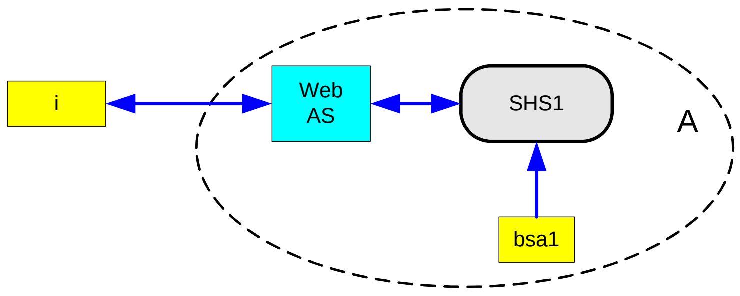
Another way to leave information to an authority is that the information supplier has a special program suited for a particular type of information. This program creates SHS-messages and supplies them to an SHS. The program resembles in large shssend (see reference [API]) apart from that is is locked to a particular product type and receiver.

In this scenario the information supplier is a small company that leaves information to an authority. The authority has published a product type. The company has its own client program that requests the information pertaining to the product type, creates an SHS-message and sends it to SHS.

The program will not use the public directory to find the address (URL) to use to connect to the SHS-service. This will be locally configured in the program.

| | |
|---|---|
| **Parties** | **i** (5515056789), a company that is not an SHS actor<br>**A** (2021000123), en actor<br>**Bsa1**, business system at **A** |
| **SHS** | **SHS1**, belongs to **A**. |
| **Product types** | **P**, a product type published by **A** |
| **Procedure** | The company **i** runs its program that collects data and produces an SHS-message. In the SHS-message the program sets *urn:X-shs:2021000123* as receiver and *ono:5515056789* as originator. The product type is **P** and the sender is left empty. The message is sent to **SHS1**.<br>In **SHS1** the receiver address will be rewritten to *urn:X-shs:2021000123.bsa1* and the sender address will be set to *urn:X-shs:2021000123*.<br>Thereafter **bsa1** can fetch the message and via the originator field see that company **i** left the information regarding product type **P**. |

### 12.2.3 Distribution of information from an authority via a mailbox.

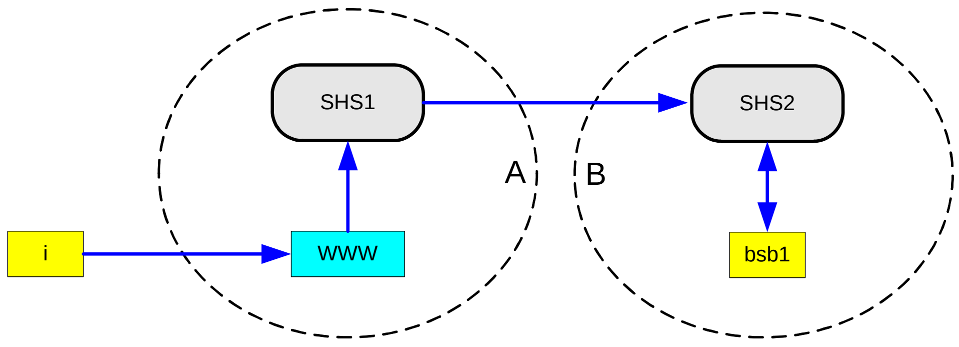


When an authority needs to send information to an individual or an organisation that are not SHS actors, an adaptation server is recommended. The adaptation server will supply the services to the individual who will not be concerned with the actual technology. An example of an adaptation service can be an electronic mailbox.

In this scenario an authority has published a product type and established electronic mailboxes (service on the web) where individuals can collect information regarding this product type that the authority has sent to the mailbox. The responsible business system sends the information as SHS-messages. Via the electronic mailboxes the individuals can list and retrieve these messages.

| | |
|---|---|
| **Parter** | **i** (7209191234), an individual<br>**A** (2021000123), an actor (authority)<br>**bsa1**, business system at **A** |
| **SHS** | **SHS1**, belongs to **A**. |
| **Product type** | **P**, a product type published by A |
| **Procedure** | Business system **bsa1** creates an SHS-message in which it assigns *urn:X-shs:2021000123.bsa1* as sender and *urn:X-shs:2021000123.ebl* as receiver. **Bsa1** also sets *pno:7209191234* as end-recipient and **P** as product type. The message is sent to **SHS1**.<br>**SHS1** sees that *urn:X-shs:2021000123.ebl* is a local recipient and prepares the message for distribution.<br>The individual **i** connects to the electronic mailbox and leaves his/her social security number. The mailbox lists the messages addressed to *urn:X-shs:2021000123.ebl* and has *pno:7209191234* as end-recipient.<br>Individual **i** can thereafter choose to retrieve the messages. The information parts contained in the SHS-message are thus made locally available for the individual who can for example write the information to a local hard disk. |

### 12.2.4 Information dispatch via a service provider



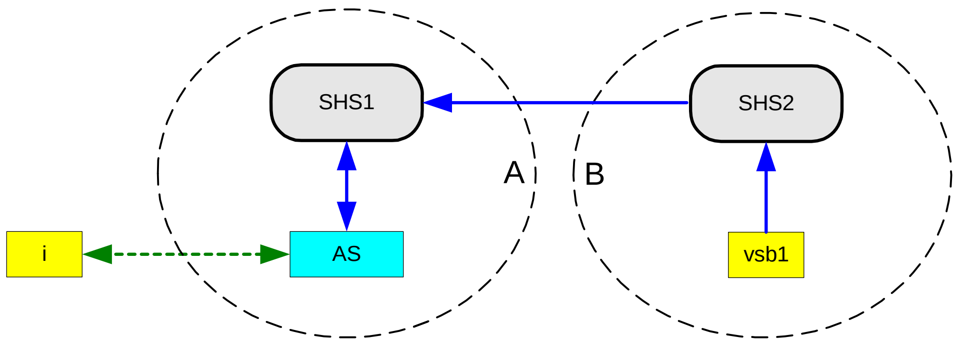
In this scenario an individual leaves information via a web service offered by a service provider. This service provider has an SHS and is thus an actor.

An authority has published a product type that can be used by other known actors.

An individual connects to the web-server of the service provider and fills in the pertinent data. The web server creates an SHS-message and sends it to his own SHS. This SHS sends the message on to the SHS of the authority and its receiving business system.

| | |
|---|---|
| **Parties** | **i** (7209191234), an individual<br>**A** (2021000123), an actor and a service provider<br>**B** (2021000124), an actor and an authority<br>**vsb1**, business system at **B** |
| **SHS** | **SHS1**, belongs to **A**<br>**SHS2**, belongs to **B** |
| **Product types** | **P**, a product type published by **B** |
| **Procedure** | Individual **i** connects to the web service of the service provider and leaves the information regarding product type **P** and his/her social security number.<br>The web server builds an SHS-message in whish it assigns *urn:X-shs:2021000123.www* as sender and leaves the receiver empty. The web service assigns *pno:7209191234* as originator and **P** as product type. The message is sent to **SHS1**.<br>**SHS1** sends this to **SHS2** based on its agreements. It assigns *urn:X-shs:2021000124* as receiver.<br>In **SHS2** the receiver address will be rewritten to *urn:X-shs:2021000124.vsb1*.<br>Thereafter **vsb1** can fetch the message and via the originator field see that individual **i** left information regarding product type **P**. |

### 12.2.5 Information dispatch via a service provider



In this scenario an authority sends information to an individual via an adaptation server at a service provider. This service provider has an SHS and is thus an actor. The adaptation server can have various methods of connecting to the individual, for example email or services on the web.

The authority has published a product type that can be used by other known actors.

The business system at the authority sends the information as an SHS-message to its own SHS. This SHS sends it further to the SHS of the service provider. The information is then made available to the individual via the adaptation server

| | |
|---|---|
| **Parties** | **i** (7209191234), an individual<br>**A** (2021000123), an actor and a service provider<br>**B** (2021000124), an actor and an authority<br>**vsb1**, business system at **B** |
| **SHS** | **SHS1**, belongs to **A**<br>**SHS2**, belongs to **B** |
| **Product types** | **P**, a product type published by **B** |
| **Procedure** | The business system **vsb1** creates an SHS-message in which it assigns *urn:X-shs:2021000124.vsb1* as sender and leaves the receiver empty. The business system also assigns *pno:7209191234* asn end-recipient and **P** as product type. The message is sent to **SHS2**.<br>**SHS2** looks at the agreements and sends the message further to **SHS1** by assigning *urn:X-shs:2021000123* as receiver.<br>In **SHS1** the receiver address will be rewritten to *urn:X-shs:2021000123.as*.<br>The adaptation server contacts **SHS1** and fetches the messages addressed to *urn:X-shs:2021000123.as* and has *pno:7209191234* as end-recipient. These messages are then made available to individual **i**.<br>At what occasion the adaptation server contacts **SHS1,** is either an initiative of individual **I,** or can be defined to occur at predetermined occasions. |