SOAP

Simple Object Access Protocol

Web services protocols stack

Web services protocols stack

Transport
HTTP, SMTP, HTTPS

Web services protocols stack

Transport
HTTP, SMTP, HTTPS

Format
XML

Web services protocols stack

Web services protocols stack

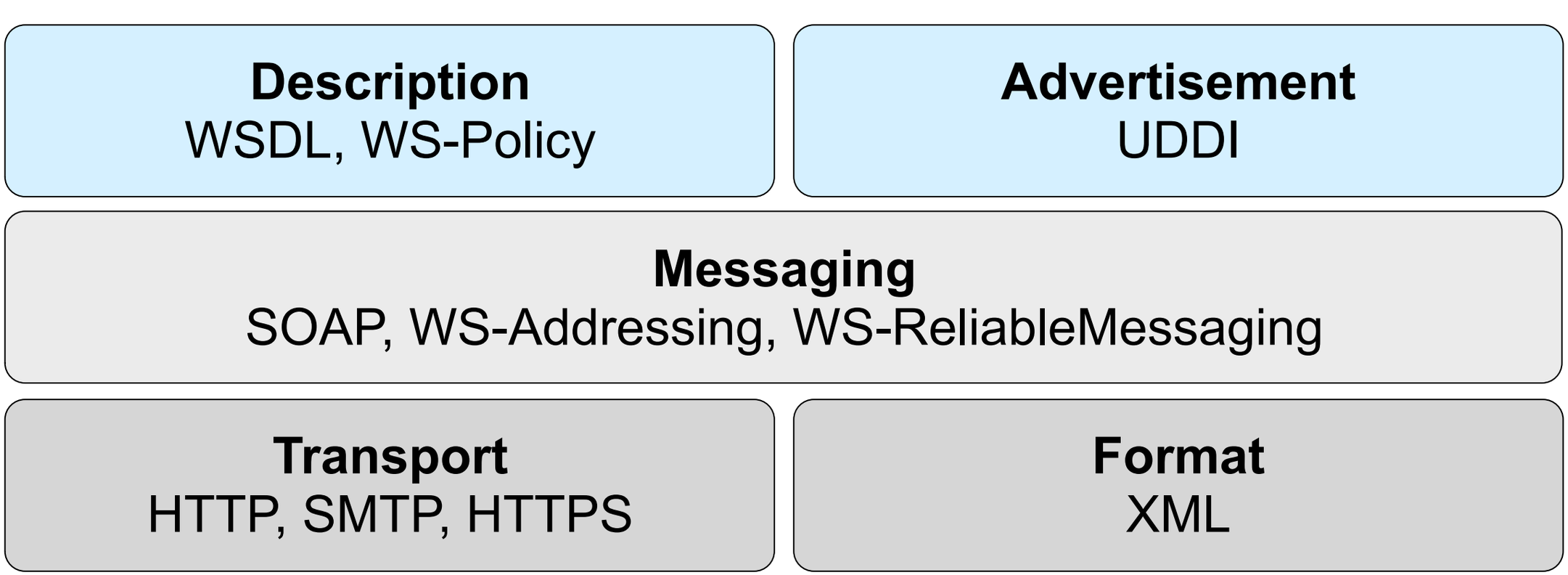
Description
WSDL, WS-Policy

Messaging
SOAP, WS-Addressing, WS-ReliableMessaging

Transport
HTTP, SMTP, HTTPS

Format
XML

Web services protocols stack



Web services protocols stack

Coordination - Context - Transactions - Security
WS-Coordination, WS-AtomicTransactions, WS-Security, …

Description
WSDL, WS-Policy

Advertisement
UDDI

Messaging
SOAP, WS-Addressing, WS-ReliableMessaging

Transport
HTTP, SMTP, HTTPS

Format
XML

Web services protocols stack

Composition - Processes
BPEL, BPELJ, WS-CDL

Coordination - Context - Transactions - Security
WS-Coordination, WS-AtomicTransactions, WS-Security, …

Description
WSDL, WS-Policy

Advertisement
UDDI

Messaging
SOAP, WS-Addressing, WS-ReliableMessaging

Transport
HTTP, SMTP, HTTPS

Format
XML

What is SOAP?

- The W3C started working on SOAP in 1999.
- Originally: Simple Object Access Protocol - Now: Service Oriented Application Protocol
- SOAP covers the following main areas:
  - Message construct: A message format for one-way communication describing how a message can be packed into an XML document
  - Processing model: rules for processing a SOAP message and a simple classification of the entities involved in processing a SOAP message. Which parts of the messages should be read by whom and how to react in case the content is not understood
  - Extensibility Model: How the basic message construct can be extended with application specific constructs
    - for security, reliability, correlation, etc.
  - Protocol binding framework: Allows SOAP messages to be transported using different protocols (HTTP, SMTP, ...)
    - A concrete binding for HTTP

Message Construct

- A mandatory extensible envelope expressing
  - what features and services are represented in a message
  - who should deal with them and whether they are optional or mandatory

- An optional set of encoding rules for data
  - For application-defined data types

- A convention for representation RPC
  - How to make calls and responses

- A protocol binding to HTTP

SOAP Envelope Anatomy

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP:Header>
      <t:Transaction xmlns:t="some-URI" SOAP:mustUnderstand="1">
            5
      </t:Transaction>
   </SOAP:Header>
   <SOAP:Body>
      <m:Deposit xmlns:m="Some-URI">
         <m:amount>200</m:amount>
      </m:Deposit>
   </SOAP:Body>
</SOAP:Envelope>
```

SOAP Envelope Anatomy

SOAP Envelope

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP:Header>
     <t:Transaction xmlns:t="some-URI" SOAP:mustUnderstand="1">
           5
     </t:Transaction>
  </SOAP:Header>
  <SOAP:Body>
     <m:Deposit xmlns:m="Some-URI">
        <m:amount>200</m:amount>
     </m:Deposit>
  </SOAP:Body>
</SOAP:Envelope>
```

SOAP Envelope Anatomy

SOAP Header
SOAP Envelope

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP:Header>
      <t:Transaction xmlns:t="some-URI" SOAP:mustUnderstand="1">
            5
      </t:Transaction>
   </SOAP:Header>
   <SOAP:Body>
      <m:Deposit xmlns:m="Some-URI">
         <m:amount>200</m:amount>
      </m:Deposit>
   </SOAP:Body>
</SOAP:Envelope>
```

SOAP Envelope Anatomy

SOAP Body

SOAP Header

SOAP Envelope

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP:Header>
      <t:Transaction xmlns:t="some-URI" SOAP:mustUnderstand="1">
            5
      </t:Transaction>
   </SOAP:Header>
   <SOAP:Body>
      <m:Deposit xmlns:m="Some-URI">
         <m:amount>200</m:amount>
      </m:Deposit>
   </SOAP:Body>
</SOAP:Envelope>
```

SOAP Envelope Anatomy

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: 299
```

HTTP

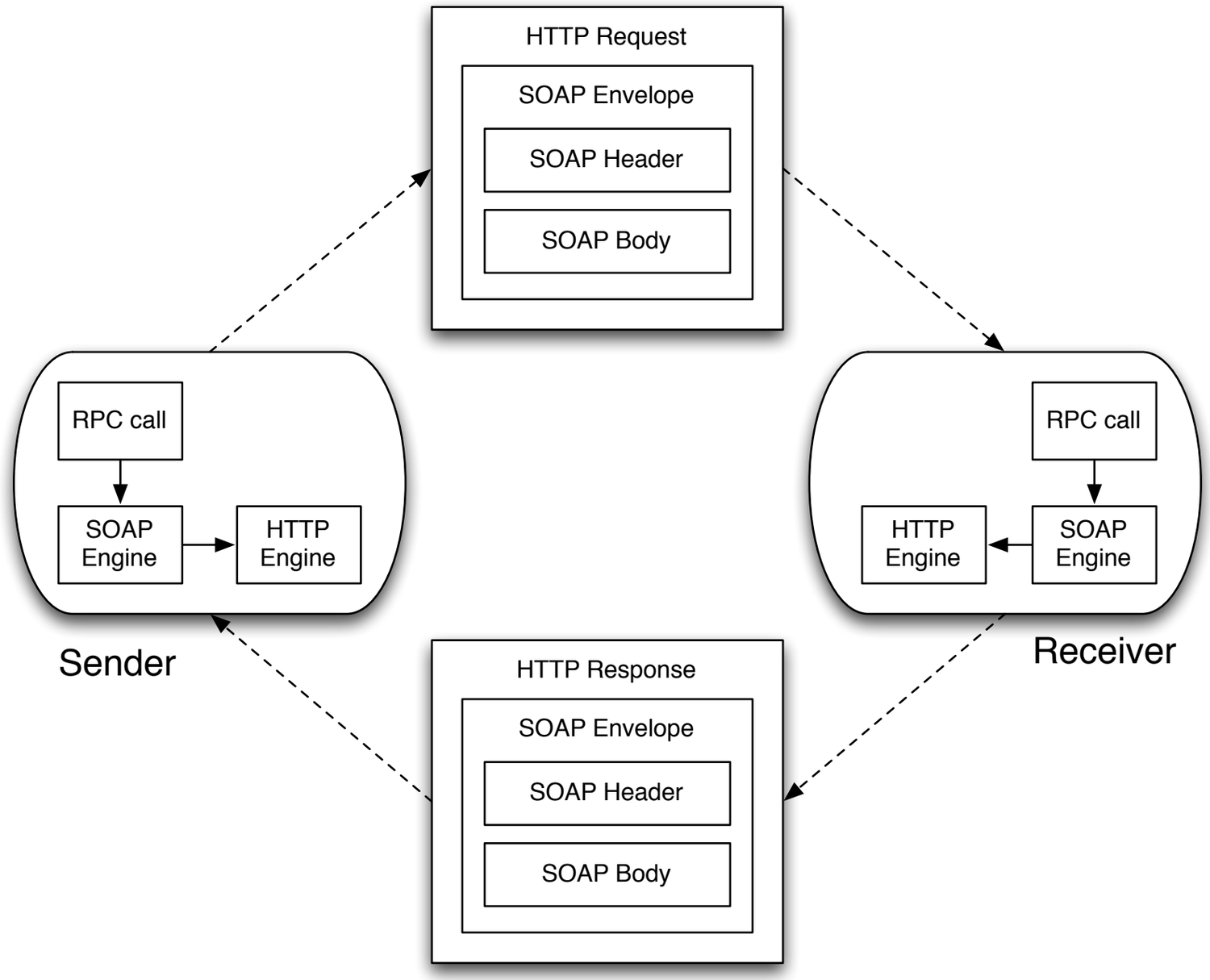
SOAP Body

SOAP Header

SOAP Envelope

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP:Header>
     <t:Transaction xmlns:t="some-URI" SOAP:mustUnderstand="1">
           5
     </t:Transaction>
  </SOAP:Header>
  <SOAP:Body>
     <m:Deposit xmlns:m="Some-URI">
        <m:amount>200</m:amount>
     </m:Deposit>
  </SOAP:Body>
</SOAP:Envelope>
```

SOAP RPC


HTTP Request
SOAP Envelope
SOAP Header
SOAP Body
RPC call
SOAP Engine
HTTP Engine
Sender
RPC call
HTTP Engine
SOAP Engine
Receiver
HTTP Response
SOAP Envelope
SOAP Header
SOAP Body

The Processing Model

- It describes a distributed processing model (with intermediary nodes)
  - Each node is identified by a unique URI

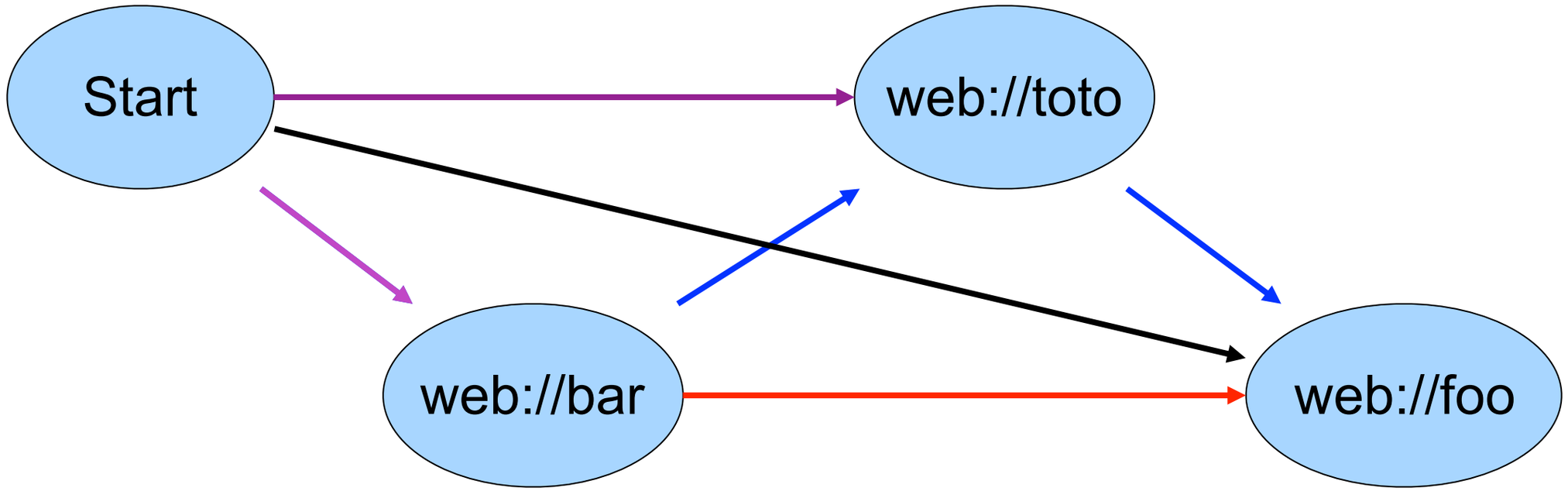
- Defines the following nodes:
  - SOAP sender: any node that sends a message
  - SOAP receiver: any node that receives a message

  - SOAP message path: the set of all nodes that witness a message's passage
  - Initial SOAP Sender: the sender that initiates a message path
  - Ultimate SOAP Receiver: the final receiver in a message path
  - SOAP intermediary: any node in a path that is not the initiale SOAP sender or the ultimate SOAP receiver

- During processing a node can have roles:
  - each role is identified by a URI (next, none, ultimateReceiver, app-specific)
  - the specification does not prescribe the criteria by which a given node determines the set of roles in which it acts on a given message
    - hard coded choices in the implementation, information provided by the underlying protocol binding, or configuration information provided by users during system installation

SOAP Headers

- Header blocks can be targeted to a specific role
  - can be mandatory or not (mustUnderstand attribute)
  - uses the role attribute
- Allows for modular addition of features and services
  - Open-ended set of headers
    - Authentication, privacy, security etc. etc.



- Routing is not part of the core standard
  - part of the extensibility model

Processing Algorithm

- Determine the set of roles in which the node is to act.
- Identify all header blocks targeted at the node that are mandatory.
- If one or more of the SOAP header blocks are not understood by the node
  - generate a single SOAP fault with the Value of Codeset to "env:MustUnderstand".
  - any further processing MUST NOT be done.
  - faults relating to the contents of the SOAP body MUST NOT be generated in this step.
- Process mandatory SOAP header blocks and, in the case of an ultimate SOAP receiver, the SOAP body.
  - A node MAY also choose to process non-mandatory SOAP header blocks targeted at it.
- In the case of a SOAP intermediary relay the message.

Faults

- One of the "problems" of distributed computing is that things can go wrong for many different reasons
  - Servers may fail, networks may go down, services may change or go away
- Need a way to communicate failures back to message originators.
  - Consider HTTP faults
- SOAP Provides its own fault communication mechanism
  - These may be in addition to HTTP errors when we use SOAP over HTTP

Fault messages

- Fault messages are included in the <body>
  - <code> (mandatory)
    - Contains one of the standard fault code enumerations
      - DataEncodingUnknown: you sent data encoded in some format that I don't understand
      - MustUnderstand: I don't support this header
      - Receiver: message was correct, but receiver could not process for some reason
      - Sender: message was incorrectly formatted, or lacked required additional information
      - VersionMismatch: I don't support your version of SOAP.
    - It may also contain subcodes for more detailed error messages
      - They don't have standard values
      - This is an extensibility mechanism
      - Subcodes may contain other subcodes

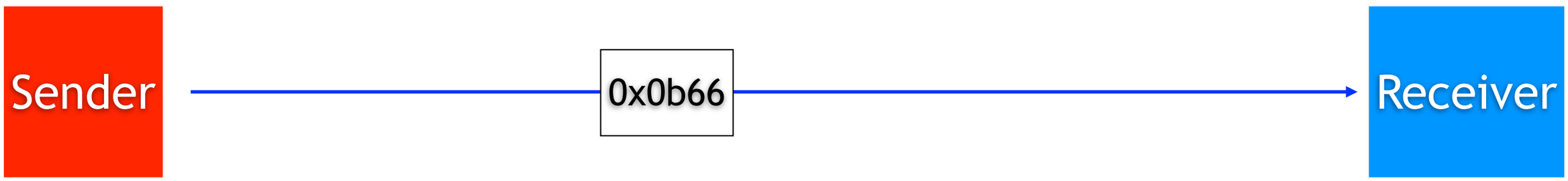
Fault messages

- <reason> (mandatory)
  - Is intended to provide human readable reasons
  - Is just a simple string determined by the implementer
- <node> and <role>
  - Are used in SOAP processing steps
- <detail> (optional)
  - Is just an extension element
  - Carries application specific information
  - It can contain any number of elements of any type

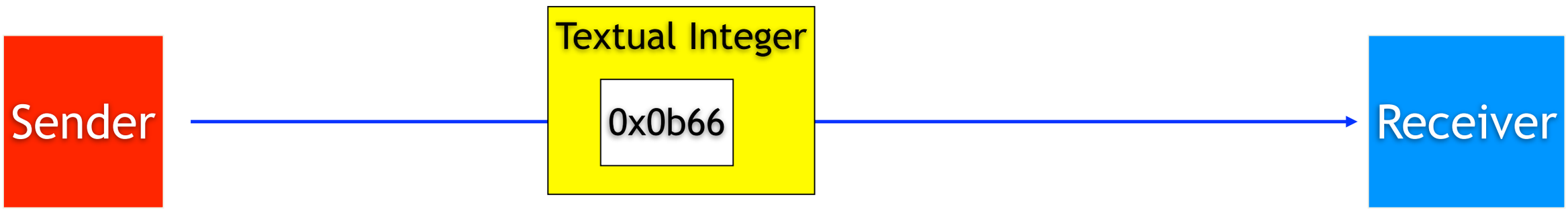
SOAP and "Binary" Data

- "Binary" can in fact mean any data that is to be tunneled through SOAP
  - Encrypted data, images, XML documents, SOAP envelopes as data
- Can be carried in two ways
  - Within the envelope as binary blob
  - Referenced from within the SOAP envelope
- References can point to anything including
  - MIME multipart, HTTP accessible resources etc.
  - Integrity can be obtained through manifest

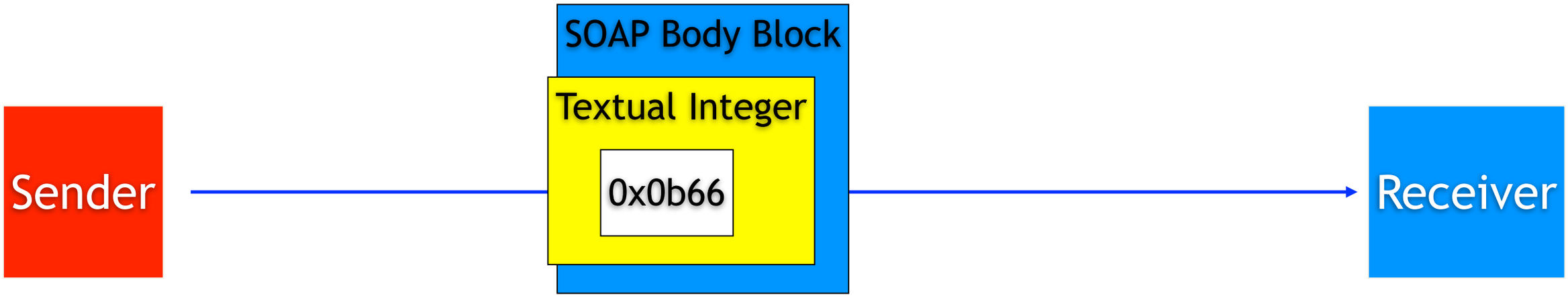
Summary



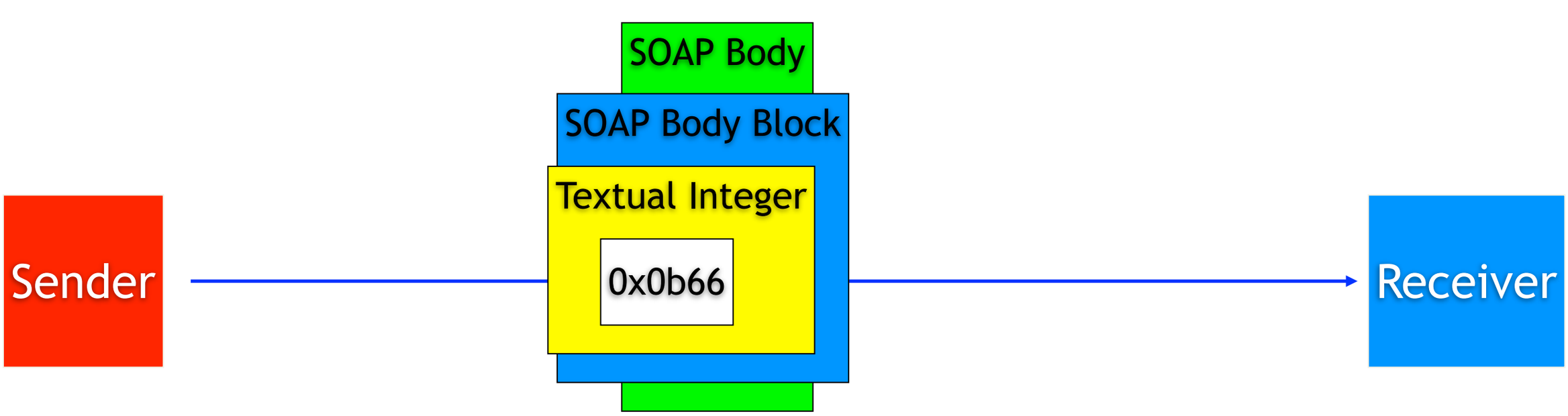
Summary



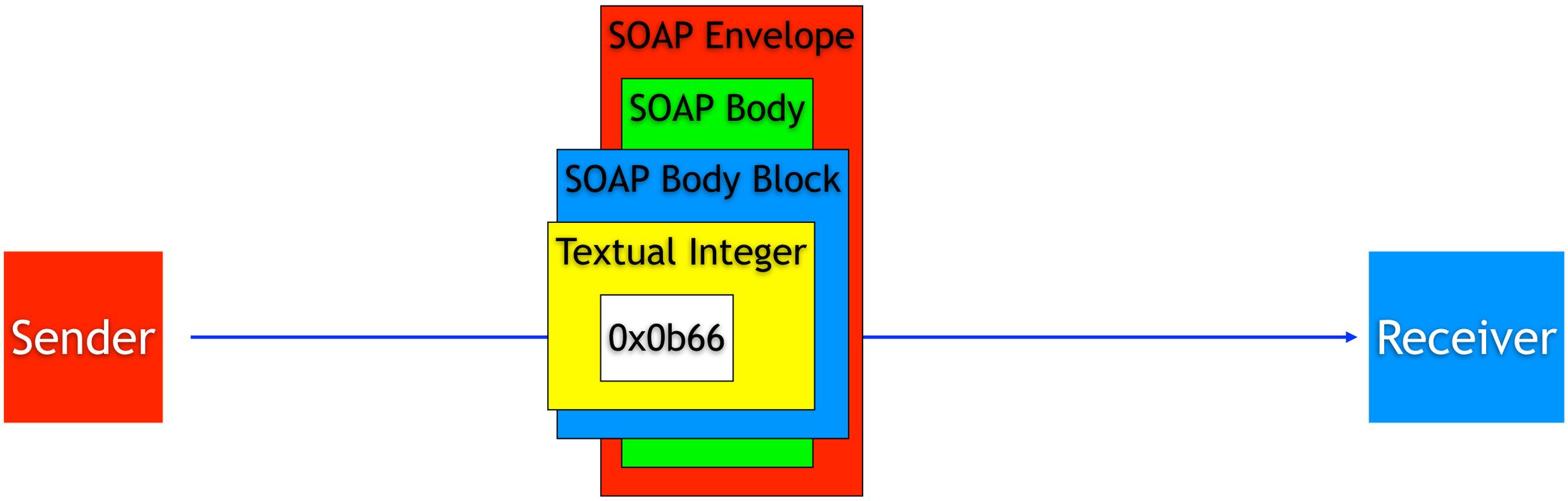
Summary



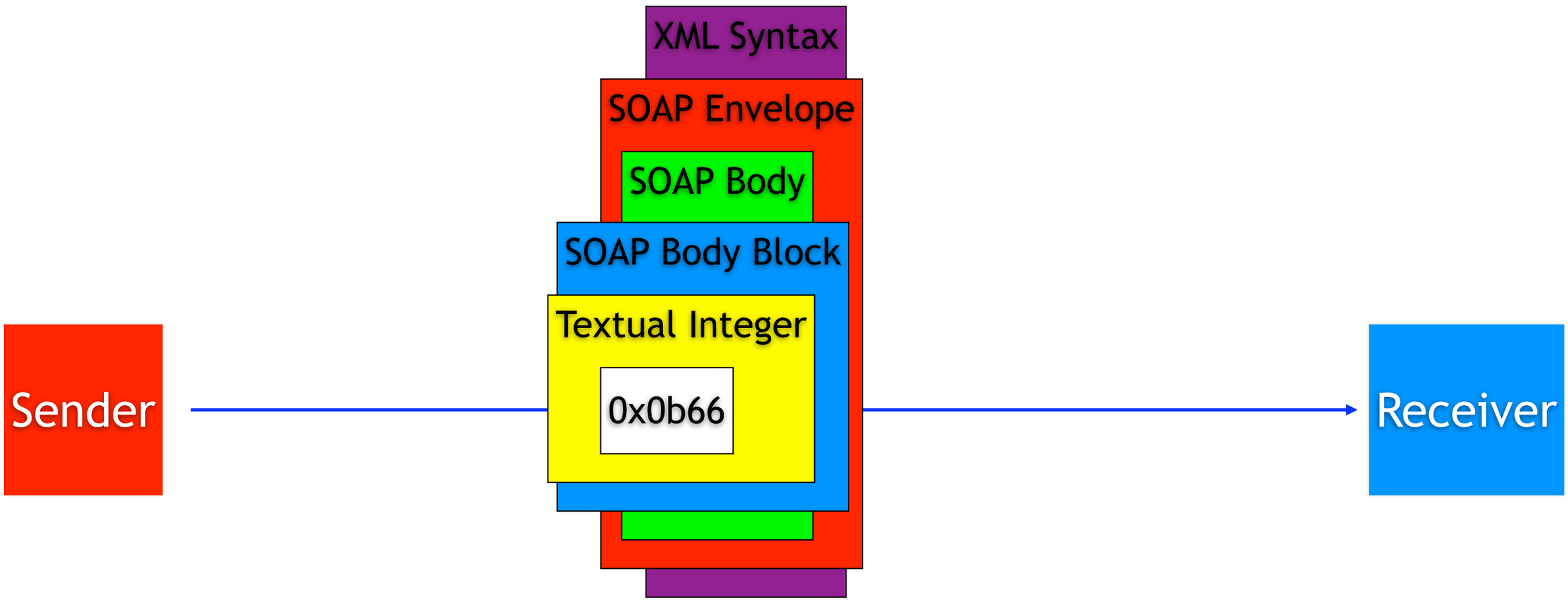
Summary


SOAP Body
SOAP Body Block
Textual Integer
0x0b66
Sender
Receiver

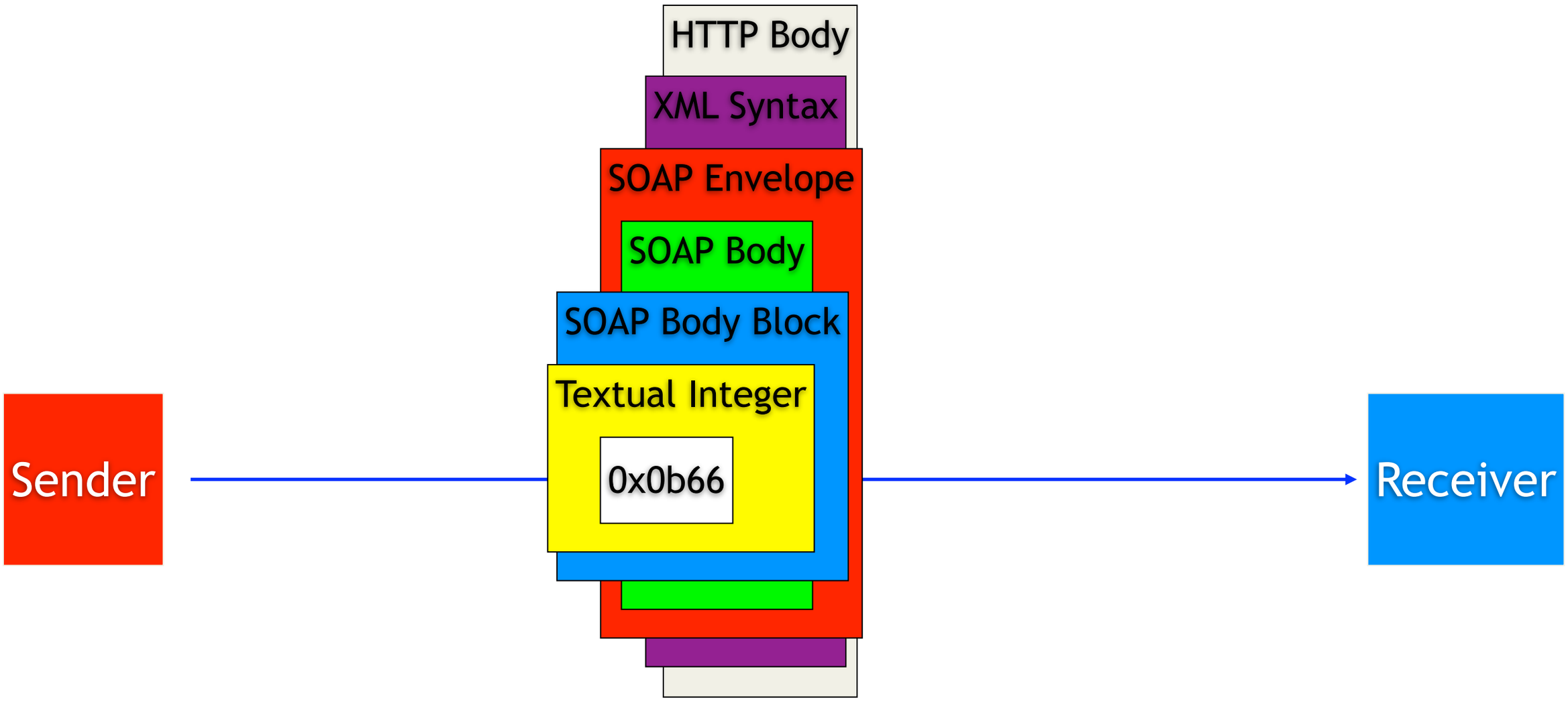
Summary


SOAP Envelope
SOAP Body
SOAP Body Block
Textual Integer
0x0b66
Sender
Receiver

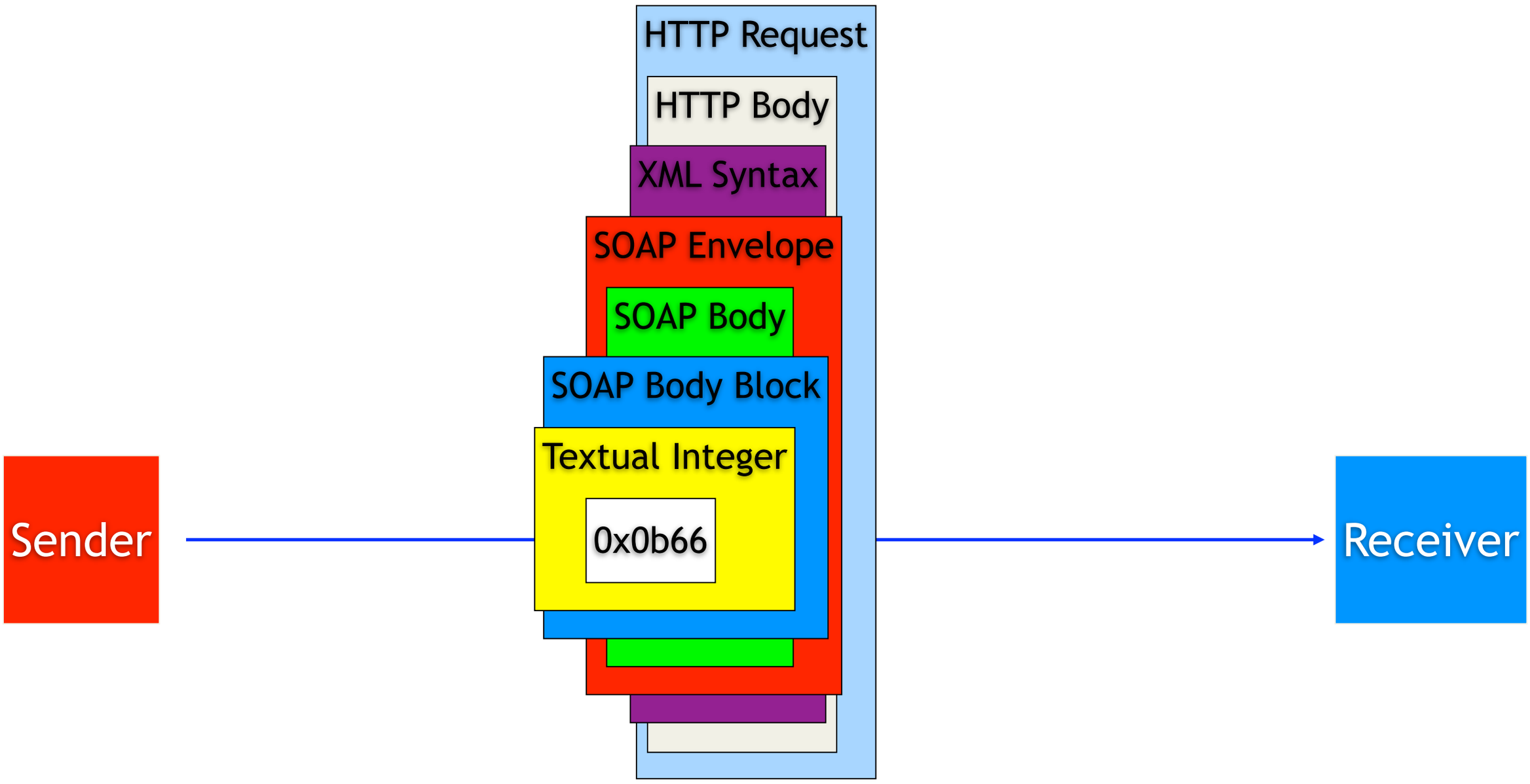
Summary


XML Syntax
SOAP Envelope
SOAP Body
SOAP Body Block
Textual Integer
0x0b66
Sender
Receiver

Summary


HTTP Body
XML Syntax
SOAP Envelope
SOAP Body
SOAP Body Block
Textual Integer
0x0b66
Sender
Receiver

Summary


HTTP Request
HTTP Body
XML Syntax
SOAP Envelope
SOAP Body
SOAP Body Block
Textual Integer
0x0b66
Sender
Receiver