Network Working Group
Request for Comments: 4227
Obsoletes: 3288
Category: Standards Track

E. O'Tuathail
Clipcode.com
M. Rose
Dover Beach Consulting, Inc.

January 2006

Using the Simple Object Access Protocol (SOAP) in Blocks Extensible Exchange Protocol (BEEP)

Status of This Memo

Copyright Notice

Abstract

This memo specifies a Simple Object Access Protocol (SOAP) binding to the Blocks Extensible Exchange Protocol (BEEP) core. A SOAP binding describes how SOAP messages are transmitted in the network.

The SOAP is an XML-based (eXtensible Markup Language) messaging protocol used to implement a wide variety of distributed messaging models. It defines a message format and describes a variety of message patterns, including, but not limited to, Remote Procedure Calling (RPC), asynchronous event notification, unacknowledged messages, and forwarding via SOAP intermediaries.

Table of Contents

1. Introduction

This memo specifies how SOAP envelopes [15] are transmitted using a BEEP profile [1]. Conforming implementations MUST support SOAP version 1.2 [15] and MAY support other versions, such as SOAP version 1.1 [17]. This memo specifies how SOAP envelopes [15] are transmitted using a BEEP profile [1]. Unlike its predecessor, RFC3288 [16], this memo does not mandate the use of SOAP version 1.1.

Throughout this memo, the term "envelope" refers to the top-level element exchanged by SOAP senders and receivers. For example, when referring to SOAP version 1.2, the term "envelope" refers to the "Envelope" element defined in Section 5.1 of [2]. Furthermore, the terms "peer", "client", "server", "one-to-one", and "one-to-many" are used in the context of BEEP. In particular, Sections 2.1 and 2.1.1 of [1] discuss BEEP roles and exchange styles.

2. BEEP Profile Identification

The BEEP profile for SOAP is identified as

    http://iana.org/beep/soap/VERSION

in the BEEP "profile" element during channel creation. where "VERSION" refers to the numeric version of the SOAP specification.

For example,

    http://iana.org/beep/soap/1.2

refers to version 1.2.

Note that RFC 3288 [16] used

    http://iana.org/beep/soap

for the purposes of profile identification for SOAP version 1.1 envelopes [17]. If an implementation of this memo chooses to implement SOAP version 1.1, then it should support both this Uniform Resource Identifier (URI) for profile identification as well as "http://iana.org/beep/soap/1.1".

In BEEP, when the first channel is successfully created, the "serverName" attribute in the "start" element identifies the "virtual host" associated with the peer acting in the server role, e.g.,

```
<start number='1' serverName='stockquoteserver.example.com'>
    <profile uri='http://iana.org/beep/soap/1.2' />
</start>
```

The "serverName" attribute is analogous to HTTP's "Host" request-header field (cf. Section 14.23 of [4]).

There are two states in the BEEP profile for SOAP, "boot" and "ready":

- In the "boot" state, the peer requesting the creation of the channel sends a "bootmsg" (either during channel initialization or in a "MSG" message).
  - If the other peer sends a "bootrpy" (either during channel initialization or in an "RPY" message), then the "ready" state is entered
  - Otherwise, the other peer sends an "error" (either during channel initialization or in an "ERR" message), then no state change occurs.
- In the "ready" state, either peer begins a SOAP message pattern by sending a "MSG" message containing an envelope. The other peer completes the message pattern either by
  - sending back an "RPY" message containing an envelope or
  - sending back zero or more "ANS" messages, each containing an envelope, followed by a "NUL" message.

  Regardless, no state change occurs.

2.1. Profile Initialization

The boot message is used for two purposes:

resource identification: each channel bound to the BEEP profile for SOAP provides access to a single resource (a network data object or service).

feature negotiation: if new features of SOAP (such as compression) emerge, their use can be negotiated.

The DTD syntax for the boot message and its response are:

```
<!ELEMENT bootmsg     EMPTY>
<!ATTLIST bootmsg
          resource    CDATA             #REQUIRED
          features    NMTOKENS          "">

<!ELEMENT bootrpy     EMPTY>
<!ATTLIST bootrpy
          features    NMTOKENS          "">
```

The boot message contains a mandatory and an optional attribute:

- the "resource" attribute, which is analogous to HTTP's "abs_path" Request-URI parameter (cf. Section 5.1.2 of [4]) and

- the "features" attribute, which, if present, contains one or more feature tokens, each indicating an optional feature of the BEEP profile for SOAP that is being requested for possible use over the channel.

Section 7.1 defines a registration template for optional features.

If the peer acting in the server role recognizes the requested resource, it replies with the boot response that contains one optional attribute:

- The "features" attribute, if present, contains a subset of the feature tokens in the boot message, indicating which features may be used over the channel. (If not present or empty, then no features may be used.)

Otherwise, if the boot message is improperly formed, or if the requested resource is not recognized, the peer acting in the server role replies with an error message (cf. Section 7.1 of [1]). Typically, the boot message and its response are exchanged during channel initialization (cf. Section 2.3.1.2 of [1]).

For example, here the boot message and its response are exchanged during channel initialization:

```
C: <start number='1' serverName='stockquoteserver.example.com'>
C:     <profile uri='http://iana.org/beep/soap/1.2'>
C:         <![CDATA[<bootmsg resource='/StockQuote' />]]>
C:     </profile>
C: </start>
```

```
    S: <profile uri='http://iana.org/beep/soap/1.2'>
    S:     <![CDATA[<bootrpy />]]>
    S: </profile>
```

The channel bound to the BEEP profile for SOAP is now in the "ready" state.

Alternatively, here is an example in which the boot exchange is unsuccessful:

```
    C: <start number='1' serverName='stockquoteserver.example.com'>
    C:     <profile uri='http://iana.org/beep/soap/1.2'>
    C:         <![CDATA[<bootmsg resource='/StockPick' />]]>
    C:     </profile>
    C: </start>

    S: <profile uri='http://iana.org/beep/soap/1.2'>
    S:     <![CDATA[<error code='550'>resource not
    S:                                 supported</error>]]>
    S: </profile>
```

Although the channel was created successfully, it remains in the "boot" state.

3. SOAP Message Packages

The BEEP profile for SOAP transmits envelopes encoded as UTF-8 and SHOULD use the media type "application/soap+xml" [5], e.g.,

```
MSG 1 1 . 0 284
Content-Type: application/soap+xml

<env:Envelope
     xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
   <m:GetLastTradePrice xmlns:m="Some-URI" />
  </env:Header>
  <env:Body>
    <symbol xmlns:p="Some-URI" >DIS</symbol>
  </env:Body>
</env:Envelope>
END
```

To provide compatibility with RFC 3288 [16], it MAY use the media type "application/xml" [6].

In addition, an implementation of the BEEP profile for SOAP MAY support transmission of envelopes using the MTOM [7] / XOP [8] packaging technique, e.g.,

```
MSG 1 2 . 283 1436
MIME-Version: 1.0
Content-Type: Multipart/Related;boundary=MIME_boundary;
    type="application/xop+xml";
    start="<mymessage.xml@example.org>";
    startinfo="application/soap+xml; action=
Content-Description: A SOAP message with my pic and sig in it

--MIME_boundary
Content-Type: application/xop+xml;
    charset=UTF-8;
    type="application/soap+xml; action=
Content-Transfer-Encoding: 8bit
Content-ID: <mymessage.xml@example.org>

<soap:Envelope
    xmlns:soap='http://www.w3.org/2003/05/soap-envelope'
    xmlns:xmlmime='http://www.w3.org/2004/11/xmlmime'>
  <soap:Body>
    <m:data xmlns:m='http://example.org/stuff'>
      <m:photo
  xmlmime:contentType='image/png'><xop:Include
    xmlns:xop='http://www.w3.org/2004/08/xop/include'
    href='cid:http://example.org/me.png'/></m:photo>
      <m:sig
  xmlmime:contentType='application/pkcs7-signature'><xop:Include
    xmlns:xop='http://www.w3.org/2004/08/xop/include'
    href='cid:http://example.org/my.hsh'/></m:sig>
    </m:data>
  </soap:Body>
</soap:Envelope>

--MIME_boundary
Content-Type: image/png
Content-Transfer-Encoding: binary
Content-ID: <http://example.org/me.png>

// binary octets for png

--MIME_boundary
Content-Type: application/pkcs7-signature
Content-Transfer-Encoding: binary
Content-ID: <http://example.org/my.hsh>
```

```
// binary octets for signature

--MIME_boundary--
END
```

Consult Section 4.1 of XOP [8] for guidance on MIME Multipart/Related usage. Because BEEP provides an 8-bit-wide path, a "transformative" Content-Transfer-Encoding (e.g., "base64" or "quoted-printable") should not be used. Note that MIME [9] requires that the value of the "Content-ID" header be globally unique. As stated in Section 4 of XOP [8], XOP may be used with diverse packaging mechanisms. When an implementation of BEEP in SOAP does support MTOM/XOP, it SHOULD support the MIME Multipart/Related XOP Package format, and MAY support others. Additional formats could, in the future, include XOP package formats specific to BEEP (e.g., sending the attachments on a different channel to the SOAP channel, which would avoid searching for the MIME boundary tags and allows lazy delivery of attachments, delivering them only when really needed.)

4. SOAP Message Patterns

4.1. One-Way Message

A one-way message involves sending a message without any response being returned.

The BEEP profile for SOAP achieves this using a one-to-many exchange, in which the client sends a "MSG" message containing an envelope, and the server immediately sends back a "NUL" message, before processing the contents of the envelope.

4.2. Request-Response Exchange

A request/response exchange involves sending a request, which results in a response being returned.

The BEEP profile for SOAP achieves this using a one-to-one exchange, in which the client sends a "MSG" message containing an envelope, and the server sends back a "RPY" message containing an envelope.

4.3. Request/N-Responses Exchange

A request/N-responses exchange involves sending a request, which results in zero or more responses being returned.

The BEEP profile for SOAP achieves this using a one-to-many exchange, in which the client sends a "MSG" message containing an envelope, and the server sends back zero or more "ANS" messages, each containing an envelope, followed by a "NUL" message.

4.4. Error Handling

The BEEP profile for SOAP does not use the "ERR" message for SOAP faults. When performing one-to-one exchanges, whatever SOAP response (including SOAP faults) generated by the server is always returned in the "RPY" message. When performing one-to-many exchanges, whatever SOAP response (including SOAP faults) generated by the server is always returned in the "ANS" messages.

If there is an error with the BEEP message unrelated to the SOAP envelope (e.g., poorly formed MIME message or MIME Content-Type not supported), then the server responds with an ERR message (see Section 7.1 of [1]) with an appropriate reply code (e.g., see Section 8 of [1]).

5. SOAP Protocol Binding Framework Conformance

5.1. Binding Name

This binding is identified by a URI that is exactly the same as the profile URI for BEEP in SOAP (see Section 2).

5.2. Base URI

The Base URI for the SOAP envelope is the URI of the resource identified in the bootmsg.

5.3. Supported SOAP Message Exchange Patterns

An implementation of this binding MUST support the following SOAP Message Exchange Pattern (MEP):

- "http://www.w3.org/2003/05/soap/mep/request-response/" (see Section 6.2 of [3])

5.4. Supported Features

An implementation of this binding MAY support the following feature: "http://www.w3.org/2003/05/soap/features/action/" (see Section 6.5 of [3].)

5.5. MEP Operation

For binding instances conforming to this specification:

- A SOAP node instantiated at the BEEP peer that initiates the message exchange may assume the role (i.e., the property http://www.w3.org/2003/05/soap/bindingFramework/ExchangeContext/Role) of "RequestingSOAPNode".

- A SOAP node instantiated at the other BEEP peer may assume the role (i.e., the property http://www.w3.org/2003/05/soap/bindingFramework/ExchangeContext/Role) of "RespondingSOAPNode".

5.5.1. Behavior of Requesting SOAP Node

The overall flow of the behavior of a requesting SOAP node follows a state machine description consistent with Section 6.2 of [3].

In order to avoid deadlock during streaming (see Section 6.2.3 of [3]), the requesting SOAP node MUST be able to process incoming SOAP response information while the SOAP request is still being transmitted.

5.5.1.1. Init

In the "Init" state, a BEEP message is formulated according to Section 3, transmission of the message begins, and then the state changes to "Requesting".

5.5.1.2. Requesting

In the "Requesting" state, more of the request message is transmitted and the arrival of the response is awaited. When the beginning of the response message is received, if it is a BEEP ERR message, then the state transitions to "Fail"; otherwise, the state transitions to "Sending+Receiving".

5.5.1.3. Sending+Receiving

In the "Sending+Receiving" state, the transmission of the request message and receiving of the response message are completed. The response message is assumed to contain a SOAP envelope serialized according to the rules for carrying SOAP messages in the media type given in the Content-Type header field. Once the receipt of the response is completed, the state transitions to "Success".

5.5.1.4. Success and Fail

"Success" and "Fail" are the terminal states for the state machine.

5.5.2. Behavior of Responding SOAP Node

The overall flow of the behavior of a responding SOAP node follows a state machine description consistent with Section 6.2 of [3]

5.5.2.1. Init

In the "Init" state, the binding awaits the start of the inbound request. In this state, it may only generate ERR messages (in accordance with Section 4.4).

5.5.2.2. Receiving

The binding begins to receive the request message and prepares the start of the response, in accordance with Section 3. When ready to transmit the response, the state transitions to "Receiving+Sending".

5.5.2.3. Receiving+Sending

The binding completes the receiving of the request and sending of the response and then transitions to "Success" state.

5.5.2.4. Success and Fail

"Success" and "Fail" are the terminal states that indicate completion of the message exchange.

6. URL Schemes

This memo defines two URL schemes, "soap.beep" and "soap.beeps", which identify the use of SOAP over BEEP over TCP. Note that, at present, a "generic" URL scheme for SOAP is not defined.

6.1. The soap.beep URL Scheme

The "soap.beep" URL scheme uses the "generic URI" syntax defined in Section 3 of [10], specifically:

- the value "soap.beep" is used for the scheme component and
- the server-based naming authority defined in Section 3.2.2 of [10] is used for the authority component.

- the path component maps to the "resource" component of the boot message sent during profile initialization (if absent, it defaults to "/").

The values of both the scheme and authority components are case-insensitive.

For example, the URL

```
soap.beep://stockquoteserver.example.com/StockQuote
```

might result in the example shown in Section 2.1.

6.1.1. Resolving IP/TCP Address Information

The "soap.beep" URL scheme indicates the use of the BEEP profile for SOAP running over TCP/IP.

If the authority component contains a domain name and a port number, e.g.,

```
soap.beep://stockquoteserver.example.com:1026
```

then the DNS is queried for the A Resource Records corresponding to the domain name, and the port number is used directly.

If the authority component contains a domain name and no port number, e.g.,

```
soap.beep://stockquoteserver.example.com
```

the Service Record algorithm [11] is used with a service parameter of "soap-beep" and a protocol parameter of "tcp" to determine the IP/TCP addressing information. If no appropriate SRV RRs are found (e.g., for "_soap-beep._tcp.stockquoteserver.example.com"), then the DNS is queried for the A RRs corresponding to the domain name and the port number used is assigned by the IANA for the registration in Section 8.4.

If the authority component contains an IP address, e.g.,

```
soap.beep://192.0.2.0:1026
```

then the DNS is not queried, and the IP address is used directly. If a port number is present, it is used directly; otherwise, the port number used is assigned by the IANA for the registration in Section 8.4.

While the use of literal IPv6 addresses in URLs is discouraged, if a literal IPv6 address is used in a "soap.beep" URL, it must conform to the syntax specified in [12].

6.2. The soap.beeps URL Scheme

The "soap.beeps" URL scheme is identical, in all ways, to the "soap.beep" URL scheme specified in Section 6.1, with the exception that prior to starting the BEEP profile for SOAP, the BEEP session must be tuned for privacy. In particular, note that both URL schemes use the identical algorithms and parameters for address resolution as specified in Section 6.1.1 (e.g., the same service name for SRV lookups, the same port number for TCP, and so on).

There are two ways to perform privacy tuning on a BEEP session, either

- a transport security profile may be successfully started or
- a user authentication profile that supports transport security may be successfully started.

Regardless, upon completion of the negotiation process, a tuning reset occurs in which both BEEP peers issue a new greeting. Consult Section 3 of [1] for an example of how a BEEP peer may choose to issue different greetings based on whether privacy is in use.

7. Registration Templates

7.1. SOAP Profile Feature Registration Template

When a feature for the BEEP profile for SOAP is registered, the following information is supplied:

Feature Identification: specify a string that identifies this feature. Unless the feature is registered with the IANA, the feature's identification must start with "x-".

Feature Semantics: specify the semantics of the feature.

Contact Information: specify the electronic contact information for the author of the feature.

8. Initial Registrations

8.1. Registration: The SOAP Profile

Profile Identification: http://iana.org/beep/soap/VERSION

Messages exchanged during Channel Creation: bootmsg, bootrpy

Messages starting one-to-one exchanges: bootmsg, a SOAP "envelope"

Messages in positive replies: bootrpy, a SOAP "envelope"

Messages in negative replies: error

Messages in one-to-many exchanges: a SOAP "envelope"

Message Syntax: a SOAP envelope

Message Semantics: corresponds to the relevant SOAP specification, e.g., for SOAP version 1.2, cf. [2].

Contact Information: Eamon O'Tuathail <eamon.otuathail@clipcode.com>, Marshall Rose <mrose@dbc.mtview.ca.us>

8.2. Registration: The soap.beep URL Scheme

URL scheme name: soap.beep

URL scheme syntax: cf. Section 6.1

Character encoding considerations: cf. the "generic URI" syntax defined in Section 3 of [10]

Intended usage: identifies a SOAP resource made available using the BEEP profile for SOAP

Applications using this scheme: cf. "Intended usage", above

Interoperability considerations: n/a

Security Considerations: cf. Section 9

Relevant Publications: cf. [2] for SOAP version 1.2

Contact Information: Eamon O'Tuathail <eamon.otuathail@clipcode.com>, Marshall Rose <mrose@dbc.mtview.ca.us>

Author/Change controller: the IESG

8.3. Registration: The soap.beeps URL Scheme

URL scheme name: soap.beeps

URL scheme syntax: cf. Section 6.2

Character encoding considerations: cf. the "generic URI" syntax defined in Section 3 of [10]

Intended usage: identifies a SOAP resource made available using the BEEP profile for SOAP after the BEEP session has been tuned for privacy

Applications using this scheme: cf. "Intended usage", above

Interoperability considerations: n/a

Security Considerations: cf. Section 9

Relevant Publications: cf. [2] for SOAP version 1.2

Contact Information: Eamon O'Tuathail <eamon.otuathail@clipcode.com>, Marshall Rose <mrose@dbc.mtview.ca.us>

Author/Change controller: the IESG

8.4. Registration: The System (Well-Known) TCP Port Number for SOAP over BEEP

Protocol Number: TCP

Message Formats, Types, Opcodes, and Sequences: cf. Section 2.1

Functions: cf. [2] for SOAP version 1.2

Use of Broadcast/Multicast: none

Proposed Name: SOAP over BEEP

Short name: soap-beep

Contact Information: Eamon O'Tuathail <eamon.otuathail@clipcode.com>, Marshall Rose <mrose@dbc.mtview.ca.us>

9. Security Considerations

Although service provisioning is a policy matter, at a minimum, all implementations MUST provide the following tuning profiles:

for authentication: http://iana.org/beep/SASL/DIGEST-MD5

for confidentiality: http://iana.org/beep/TLS (using the TLS_RSA_WITH_AES_EDE_CBC_SHA cipher)

for both: http://iana.org/beep/TLS (using the
TLS_RSA_WITH_AES_EDE_CBC_SHA cipher supporting client-side
certificates)

Furthermore, implementations may choose to offer MIME-based security services providing message integrity and confidentiality, such as OpenPGP [13] or S/MIME [14].

Regardless, consult [1]'s Section 9 for a discussion of BEEP-specific security issues.

10. IANA Considerations

Previously, the IANA registered "http://iana.org/beep/soap" for use with RFC 3288 [16]. This memo requires that the IANA register a URI-prefix of

http://iana.org/beep/soap/VERSION

to correspond to the family of profiles defined Section 8.1.

The IANA has registered "soap.beep" and "soap.beeps" as URL schemes, as specified in Section 8.2 and Section 8.3, respectively.

The IANA has also registered "SOAP over BEEP" as a TCP port number, as specified in Section 8.4.

The IANA now broadens these three registries to support the family of BEEP profiles defined by this URI prefix.

Finally, the IANA maintains a list of SOAP profile features, cf. Section 7.1. The IESG is responsible for assigning a designated expert to review the specification prior to the IANA making the assignment. Prior to contacting the IESG, developers of SOAP profile features must use the mailing list beepwg@lists.beepcore.org to solicit commentary.

11. Changes from RFC 3288

This memo differs from RFC 3288 [16] in one substantive way: a URL prefix is defined to support a family of BEEP profiles corresponding to different versions of SOAP. Similarly, the IANA registrations in Section 8.1, Section 8.3, and Section 8.4 are updated to reflect this broadening.

Support for W3C MTOM/XOP packaging has been added.

A new section was added to discuss the distributed state machine of the Request-Response MEP.

In non-substantive ways, a small number of typographical errors were corrected.

12. Acknowledgements

The authors gratefully acknowledge the contributions of: Christopher Ferris, Huston Franklin, Alexey Melnikov, Bill Mills, and Roy T. Fielding.

13. References

13.1. Normative References

[1] Rose, M., "The Blocks Extensible Exchange Protocol Core", RFC 3080, March 2001.

[2] Nielsen, H., Mendelsohn, N., Gudgin, M., Hadley, M., and J. Moreau, "SOAP Version 1.2 Part 1: Messaging Framework", W3C REC REC-soap12-part1-20030624, June 2003.

[3] Nielsen, H., Hadley, M., Moreau, J., Mendelsohn, N., and M. Gudgin, "SOAP Version 1.2 Part 2: Adjuncts", W3C REC REC-soap12-part2-20030624, June 2003.

[4] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[5] Baker, M. and M. Nottingham, "The "application/soap+xml" media type", RFC 3902, September 2004.

[6] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", RFC 3023, January 2001.

[7] Nottingham, M., Mendelsohn, N., Gudgin, M., and H. Ruellan, "SOAP Message Transmission Optimization Mechanism", W3C REC REC-soap12-mtom-20050125, January 2005.

[8] Nottingham, M., Mendelsohn, N., Gudgin, M., and H. Ruellan, "XML-binary Optimized Packaging", W3C REC REC-xop10-20050125, January 2005.

[9] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[10] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[11] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[12] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[13] Elkins, M., Del Torto, D., Levien, R., and T. Roessler, "MIME Security with OpenPGP", RFC 3156, August 2001.

[14] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004.

13.2. Informative References

[15] Mitra, N., "SOAP Version 1.2 Part 0: Primer", W3C REC REC-soap12-part0-20030624, June 2003.

[16] O'Tuathail, E. and M. Rose, "Using the Simple Object Access Protocol (SOAP) in Blocks Extensible Exchange Protocol (BEEP)", RFC 3288, June 2002.

[17] Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H., Thatte, S., and D. Winer, "Simple Object Access Protocol (SOAP) 1.1", W3C NOTE NOTE-SOAP-20000508, May 2000.

[18] Levinson, E., "The MIME Multipart/Related Content-type", RFC 2387, August 1998.

[19] Barton, J., Thatte, S., and H. Nielsen, "SOAP Messages with Attachments", W3C NOTE NOTE-SOAP-attachments-20001211, December 2000.

[20] Levinson, E., "Content-ID and Message-ID Uniform Resource Locators", RFC 2392, August 1998.

[21] Palme, J., Hopmann, A., and N. Shelness, "MIME Encapsulation of Aggregate Documents, such as HTML (MHTML)", RFC 2557, March 1999.

Appendix A. SOAP with Attachments (Informative)

To provide compatibility with RFC3288 [16], a BEEP profile for SOAP MAY allow envelopes to be transmitted as the root part of a "multipart/related" [18] content, and with subordinate parts referenced using the rules of Section 3 of [19] (i.e., using either the "Content-ID:" [20] or "Content-Location:" [21] headers), e.g.,

```
 MSG 1 2 . 278 657
 Content-Type: multipart/related; boundary="MIME_boundary";
               type=application/xml;
               start="<claim061400a.xml@claiming-it.com>"

 --MIME_boundary
 Content-Type: application/xml
 Content-ID: <claim061400a.xml@claiming-it.com>

 <?xml version='1.0' ?>
 <env:Envelope
      xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  ..
 </env:Header>
 <env:Body>
 <theSignedForm href="cid:claim061400a.tiff@claiming-it.com" />
  ..
 </env:Body>
 </env:Envelope>

 --MIME_boundary
 Content-Type: image/tiff
 Content-Transfer-Encoding: binary
 Content-ID: <claim061400a.tiff@claiming-it.com>

  ...binary TIFF image...
 --MIME_boundary--
 END
```

Consistent with Section 2 of [19], it is strongly recommended that the multipart contain a "start" parameter, and that the root part contain a "Content-ID:" header. However, because BEEP provides an 8bit-wide path, a "transformative" Content-Transfer-Encoding (e.g., "base64" or "quoted-printable") should not be used. Further note that MIME [9] requires that the value of the "Content-ID" header be globally unique.

Authors' Addresses

Eamon O'Tuathail
Clipcode.com
24 Thomastown Road
Dun Laoghaire
Dublin
IE

Phone: +353 1 2350 424
EMail: eamon.otuathail@clipcode.com
URI: http://www.clipcode.com/

Marshall T. Rose
Dover Beach Consulting, Inc.
POB 255268
Sacramento, CA 95865-5268
US

Phone: +1 916 483 8878
EMail: mrose@dbc.mtview.ca.us

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).