

FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS

FIPA Agent Message Transport Envelope Representation in String Specification

Document title	FIPA Agent Message Transport Envelope Representation in String Specification		
Document number	DC00073D	Document source	FIPA Agent Management
Document status	Deprecated	Date of this status	2001/08/10
Supersedes	FIPA00024		
Contact	fab@fipa.org		
Change history			
2000/07/20	Deprecated since current MTPs no longer use string based envelopes		
2001/08/10	Line numbering added		

© 2000 Foundation for Intelligent Physical Agents - <http://www.fipa.org/>

Geneva, Switzerland

Notice

Use of the technologies described in this specification may infringe patents, copyrights or other intellectual property rights of FIPA Members and non-members. Nothing in this specification should be construed as granting permission to use any of the technologies described. Anyone planning to make use of technology covered by the intellectual property rights of others should first obtain permission from the holder(s) of the rights. FIPA strongly encourages anyone implementing any part of this specification to determine first whether part(s) sought to be implemented are covered by the intellectual property of others, and, if so, to obtain appropriate licenses or other permission from the holder(s) of such intellectual property prior to implementation. This specification is subject to change without notice. Neither FIPA nor any of its Members accept any responsibility whatsoever for damages or liability, direct or consequential, which may result from the use of this specification.

19 **Foreword**

20 The Foundation for Intelligent Physical Agents (FIPA) is an international organization that is dedicated to promoting the
21 industry of intelligent agents by openly developing specifications supporting interoperability among agents and agent-
22 based applications. This occurs through open collaboration among its member organizations, which are companies and
23 universities that are active in the field of agents. FIPA makes the results of its activities available to all interested parties
24 and intends to contribute its results to the appropriate formal standards bodies.

25 The members of FIPA are individually and collectively committed to open competition in the development of agent-
26 based applications, services and equipment. Membership in FIPA is open to any corporation and individual firm,
27 partnership, governmental body or international organization without restriction. In particular, members are not bound to
28 implement or use specific agent-based standards, recommendations and FIPA specifications by virtue of their
29 participation in FIPA.

30 The FIPA specifications are developed through direct involvement of the FIPA membership. The status of a
31 specification can be either Preliminary, Experimental, Standard, Deprecated or Obsolete. More detail about the process
32 of specification may be found in the FIPA Procedures for Technical Work. A complete overview of the FIPA
33 specifications and their current status may be found in the FIPA List of Specifications. A list of terms and abbreviations
34 used in the FIPA specifications may be found in the FIPA Glossary.

35 FIPA is a non-profit association registered in Geneva, Switzerland. As of January 2000, the 56 members of FIPA
36 represented 17 countries worldwide. Further information about FIPA as an organization, membership information, FIPA
37 specifications and upcoming meetings may be found at <http://www.fipa.org/>.

38 **Contents**

39	1	Scope	1
40	2	String Envelope Representation	2
41	2.1	Component Name.....	2
42	2.2	Lexical Analysis	2
43	2.3	Syntax.....	3
44	2.4	Additional Syntax Rules.....	5
45	2.5	Representation of Time	5
46	3	References.....	6
47			

47 **1 Scope**

48 This document is part of the FIPA specifications and deals with message transportation between inter-operating agents.

49 This document also forms part of the FIPA Agent Management Specification [FIPA00023] and contains specifications
50 for:

51
52 Syntactic representation of a message envelope in string form.

53

54

54 2 String Envelope Representation

55 This section gives the concrete syntax for the message envelope in string format. This concrete syntax and lexical
56 analysis of messages has been inspired by [RFC822].
57

58 2.1 Component Name

59 The name assigned to this component is:

```
60  
61 fipa.mts.env.rep.string.std  
62
```

63 2.2 Lexical Analysis

64 Messages consist of message envelope parameters and, optionally, a message body. The message body is simply a
65 sequence of ASCII characters representing an ACL message. The message body is separated from the message
66 envelope by two subsequent CRLF tokens with nothing in between the tokens (that is, a line with nothing preceding the
67 CRLF).
68

69 Each message envelope parameter can be viewed as a single, logical line of ASCII characters, comprising a parameter
70 name and a parameter value. For convenience, the parameter value portion of this conceptual entity can be split into a
71 multiple-line representation by inserting, at the transmitter side, a CRLF immediately followed by at least one LWSP-
72 char (this action is called *folding*). At the receiver side, CRLF immediately followed by a LWSP-char is considered
73 equivalent to the LWSP-char (this action is called *unfolding*).
74

75 Once a parameter has been unfolded, at the receiver side it may be viewed as being composed of a parameter name,
76 followed by a colon (:), followed by a parameter body, and terminated by a carriage-return/line-feed (CRLF). The
77 parameter name must be composed of printable ASCII characters (that is, characters that have values between 33 and
78 126 decimal, except colon). The parameter body may be composed of any ASCII characters, except CR or LF. (While
79 CR and/or LF may be present in the actual text, they are removed by the action of unfolding the parameter.)
80

81 Except as noted, alphabetic strings may be represented in any combination of upper and lower case. However, ACCs
82 are required to preserve case information when transporting messages.
83

84 These rules show a parameter meta-syntax, without regard for the particular type or internal syntax. Their purpose is to
85 permit detection of parameters; also, they present to higher-level parsers an image of each parameter as fitting on one
86 line.
87

```
88 MessageEnvelope      = Parameter+ CRLF MessageBody.  
89  
90 MessageBody         = Text* ( CRLF Text* )*  
91                       | Byte*.1  
92  
93 Parameter           = ParameterName ":" [ ParameterBody ] CRLF.  
94  
95 ParameterName       = 1* <any CHAR, excluding CTLs, SPACE, and ":">.  
96  
97 ParameterBody       = ParameterBodyContents [CRLF LWSP-char ParameterBody].  
98  
99 ParameterBodyContents = <the ASCII characters making up the ParameterBody, as defined  
100                       in the following section and consisting of combinations of  
101                       Atom, QuotedString and specials tokens or else consisting of  
102                       Text>.  
103  
104
```

¹ Note that this cannot be transmitted over [FIPA00075].

104 The following rules are used to define an underlying lexical analyser, which feeds tokens to higher-level parsers.
 105
 106 ; (Octal, Decimal.)
 107
 108 CHAR = <any ASCII character>. ; (0-177, 0.-127.)
 109
 110 DIGIT = <any ASCII decimal digit>. ; (60- 71, 48.- 57.)
 111
 112 CTL = <any ASCII control ; (0- 37, 0.- 31.)
 113 character and DEL>. ; (177, 127.)
 114
 115 CR = <ASCII CR, carriage return>. ; (15, 13.)
 116
 117 LF = <ASCII LF, linefeed>. ; (12, 10.)
 118
 119 SPACE = <ASCII SP, space>. ; (40, 32.)
 120
 121 HTAB = <ASCII HT, horizontal-tab>. ; (11, 9.)
 122
 123 <"> = <ASCII quote mark>. ; (42, 34.)
 124
 125 CRLF = CR LF.
 126
 127 LWSPChar = SPACE / HTAB. ; semantics = SPACE
 128
 129 LinearWhiteSpace = ([CRLF] LWSPChar)+. ; semantics = SPACE
 130 ; CRLF => folding
 131
 132 Text = <any CHAR including bare CR and
 133 bare LF but NOT including CRLF>.
 134
 135 Atom = <any CHAR except ">, SPACE and CTLs>
 136 <any CHAR except SPACE and CTLs> *.
 137
 138 QuotedString = <"> (QText/QuotedPair)* <">. ; Regular qtext or
 139 ; quoted chars.
 140
 141 QText = <any CHAR excepting ">, ; => may be folded
 142 "\" and CR, and including linear-white-space>.
 143
 144 QuotedPair = "\" CHAR. ; may quote any char
 145
 146 Word = Atom / QuotedString.
 147
 148 Byte = <any 8-bit byte>.
 149

150 2.3 Syntax

151 The following rules apply after the unfolding operation, as specified in the previous section.

152
 153 MessageEnvelope = Parameter+ CRLF MessageBody.
 154
 155 Parameter = ACLRepresentationParameter CRLF
 156 | CommentParameter CRLF
 157 | ContentLengthParameter CRLF
 158 | ContentEncodingParameter CRLF
 159 | DateParameter CRLF
 160 | EncryptedParameter CRLF
 161 | IntendedReceiverParameter CRLF
 162 | ReceivedParameter CRLF
 163 | EnvSenderParameter CRLF
 164 | EnvReceiverParameter CRLF

165		TransportBehaviourParameter	CRLF
166		UserDefinedParameter	CRLF.
167			
168			
169	MessageBody	= Text* (CRLF Text*)*	
170		CRLF Byte*. ²	
171			
172	ACLRepresentationParameter	= "ACL-representation" ":" word.	
173			
174	CommentParameter	= "Comments" ":" text*.	
175			
176	ContentLengthParameter	= "Content-length" ":" DIGIT+.	
177			
178	ContentEncodingParameter	= "Content-encoding" ":" word.	
179			
180	DateParameter	= "Date" ":" DateTime.	
181			
182	DateTime	= See section 2.5.	
183			
184	EncryptedParameter	= "Encrypted" ":" word [word].	
185			
186	IntendedReceiverParameter	= "Intended-receiver" ":" AgentIdentifierList.	
187			
188	AgentIdentifierList	= AgentIdentifier ["," AgentIdentifier]*.	
189			
190	ReceivedParameter	= "Received" ":"	
191		["from" URL]	
192		["by" URL]	
193		["id" word]	
194		["via" word]	
195		;" DateTime.	
196			
197	EnvSenderParameter	= "From" ":" AgentIdentifier.	
198			
199	EnvReceiverParameter	= "To" ":" AgentIdentifierList.	
200			
201	TransportBehaviourParameter	= "Transport-behaviour" ":"	
202		["error-messages" AgentIdentifierList]	
203		["delivery" word]	
204		["acknowledgement" AgentIdentifierList].	
205			
206	UserDefinedParameter	= <any parameter which has not been defined in this	
207		specification or published as an extension to this	
208		specifications; parameter name must be unique and may	
209		be pre-empted by published extensions.>.	
210			
211	AgentIdentifier	= "(" "AID"	
212		":name" Word	
213		[":addresses" URLSequence]	
214		[":resolvers" AgentIdentifierSequence]	
215		(UserDefinedParameter Expression)*)".	
216			
217	AgentIdentifierSequence	= "(" "sequence" AgentIdentifier*)". ³	
218			
219	URLSequence	= "(" "sequence" URL*)".	
220			
221	URL	= See [RFC2396]	
222			

² Note that this cannot be transmitted over [FIPA00075].

³ A sequence is considered to have a left to right (first to last) ordering.

2.4 Additional Syntax Rules

The following additional rules not specified in the grammar also apply:

1. The abstract syntax of the message envelope is mandatory.

2. This specification permits multiple occurrences of message envelope parameters. For the purposes of disambiguation the first occurrence overrides any subsequent occurrence (see [RFC822] for further details).

In the future, additional parameters may be defined and added to the message envelope. Such parameters are prefixed with `X-FIPA-` and their behaviour is not specified. If an organisation wishes to add its own message envelope parameters it is suggested they prefix the new parameter name with `X-CompanyName-` to reduce the chances of conflict.

2.5 Representation of Time

Time tokens are based on [ISO8601], with extensions for relative time and millisecond durations. Time expressions may be absolute, or relative to the current time. Relative times are distinguished by the character `+` appearing as the first character in the construct. If no type designator is given, the local time zone is used. The type designator for UTC is the character `Z`. UTC is preferred to prevent time zone ambiguities. Note that years must be encoded in four digits. As examples, 8:30am on April 15th, 1996 local time would be encoded as:

```
19960415T083000000
```

The same time in UTC would be:

```
19960415T083000000Z
```

While one hour, 15 minutes and 35 milliseconds from now would be:

```
+000000000T011500035
```


253 3 References

- 254 [FIPA00023] FIPA Agent Management Specification. Foundation for Intelligent Physical Agents, 2000.
255 <http://www.fipa.org/specs/fipa00023/>
- 256 [FIPA00067] FIPA Agent Message Transport Service Specification. Foundation for Intelligent Physical Agents, 2000.
257 <http://www.fipa.org/specs/fipa00067/>
- 258 [FIPA00075] FIPA Agent Message Transport Protocol for IOP Specification. Foundation for Intelligent Physical
259 Agents, 2000.
260 <http://www.fipa.org/specs/fipa00075/>
- 261 [ISO8601] Date Elements and Interchange Formats, Information Interchange-Representation of Dates and Times.
262 International Standards Organisation, 1998.
263 <http://www.iso.ch/cate/d15903.html>
- 264 [RFC822] Uniform Resource Identifiers: Generic Syntax. Request for Comments, 1992.
265 <http://www.ietf.org/rfc/rfc0822.txt>
- 266 [RFC2396] Standard for the Format of APRA Internet Text Messages. Request for Comments, 1998.
267 <http://www.ietf.org/rfc/rfc2396.txt>