EDGE MESSAGE PROTOCOL SPECIFICATION

Standard
S-9354

Adopted: 2010

TABLE OF CONTENTS

1.0 INTRODUCTION

1.1 Purpose

This document specifies the Edge Message Protocol (EMP) message envelope. It describes the fields of the envelope, their semantics, and commentary about their use. Sufficient detail for implementation is provided.

1.2 Document Conventions

This document uses the following conventions:

- Values prefixed with “0x” are hexadecimal values.
- Code segments are displayed in Times New Roman.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers of applications that use EMP to communicate or distribute data, developers of EMP application gateways, and designers and administrators of infrastructures that support EMP.

1.4 Protocol Scope

EMP is an application message envelope used in various interoperable applications including interoperable train control (ITC). EMP is also supported by the ITC interoperable messaging architecture.

2.0 OVERALL DESCRIPTION

2.1 Protocol Perspective

The Edge Messaging Protocol (EMP) defines the message envelope (header, footer) used to communicate between the various interoperable train control (ITC) applications. EMP is also used by interoperable non-train control applications (e.g., energy management, LIG). The use of a common envelope facilitates an interoperable infrastructure for message transmission, reception, decoding, and routing.

EMP messages are typically translated into specific implementations (e.g., XML, Tibco, Oracle, ESB) when they enter a railroad’s environment. The EMP envelope contains sufficient information to allow the mapping of EMP messages into and out of the various messaging environments in a code-independent way.

2.2 Protocol Functions

2.2.1 Data Integrity

Because EMP is used as an end-to-end application level (layer 7) message format, it includes a data integrity field to protect the contents of the EMP header and payload. If intermediate nodes modify the contents of the EMP header or payload (e.g., compress, encode), they are expected to faithfully undo the modification prior to delivering the message to an interoperable node or application. The data integrity value will indicate if the EMP message header and payload delivered to the final destination are not exactly the same as the EMP header and payload generated by the message source. Safety critical applications will use an application-specific data integrity value (e.g., HMAC) in place of the CRC. If bandwidth optimizations do not faithfully recreate the input, the data integrity value will catch such occurrences. Although such occurrences would not impact safety, they may impact operations/productivity.

The message format in EMP allows for data integrity information (CRC or application-specific value such as HMAC) at the end of the messages. An option flag allows sources not wishing to calculate a data integrity value to opt out. When opting out of data integrity calculation, it is assumed that the underlying network(s) will handle data integrity issues correctly and consistently. The CRC must guard against corrupt messages with a high degree of accuracy (e.g., $< 10^{-9}$) for large messages.

Some applications may require a greater level of data integrity and message authenticating than that supported by the CRC calculation specified in paragraph 3.6, “CRC Calculation.” In such situations, the data integrity field of the EMP header can be used to carry an application-specific data integrity value (e.g., MD5, HMAC), and the application would set the appropriate bits in the flag field (see paragraph 3.2.1.4, “Flags”).

2.2.2 Byte Ordering

Multi-byte binary fields in the EMP header will use big-endian format.

2.2.3 Multiple Transport Protocols

The EMP message format is intended to support a wide range of devices and applications and the use of multiple message transport protocols. This will minimize application rewrites and reduce maintenance costs when protocols are modified or extended. This allows a system to continue to operate, albeit in a degraded mode, in case of a configuration-management error.

2.2.4 Fragmentation

The EMP message format allows for a maximum size of 16 MB. The EMP specification does not define a fragmentation protocol. If fragmentation is required, it must be implemented by the application or by the underlying messaging system.

2.3 Mapping of Protocol to OSI Layers

EMP is an application layer protocol and corresponds to the application layer of the OSI stack.

2.4 Relationship to Other Protocols

As an application layer protocol, EMP relies on other protocols to provide the functionality of lower layers of the OSI stack.

2.5 Assumptions and Dependencies

EMP assumes that a message transport (e.g., UDP, TCP, Class D, JMS) will be used to move EMP messages back and forth between applications.

3.0 PROTOCOL FORMAT

An EMP message contains two parts, the EMP wrapper and the message body. The EMP wrapper contains a header and a data integrity value. The application constructs the header, calculates the data integrity value, and packages the header, data, and data integrity value into an EMP message. The EMP wrapper is binary to facilitate efficient use of narrowband wireless networks. The EMP payload or body can be of any format (e.g., binary, XML).

3.1 Message Header Overview

Each EMP message shall contain the header shown in Table 3.1, "EMP header summary," which uses the following color-coding scheme to differentiate between the major sections of the header:

1. Common header	This is header information that every sender will complete. This information facilitates message decoding.
2. Optional header	Information that some senders will not support and will assign default values. This information is used by more advanced devices to aid in troubleshooting and auditing. These fields must contain a value, but it can be a default value indicating a lack of support for the functionality. Support of these fields is optional for the sender but *not* for the receiver. This implies that if a sender populates an optional field, the receiver must support its implementation.
3. Variable header	Some, or all, of this information may be omitted by some senders. This information is used when a message is to be routed/bridged into, or out of, a specific messaging architecture (including wireless transports). The "variable header size" field preceding this part of the header specifies its size primarily as an aid to programmers parsing the message. If this portion of the header is included, all fields must be present, although the QoS and TTL may be blank (zero) and/or the source and destination addresses may be null.

Table 3.1 EMP header summary

EMP Header		
Header Field	**Field Size (bits)**	**Definition and Notes**
Protocol (header) Version	8	Version of EMP header Each version of the EMP message format specification will define the header version number applicable to that version of the specification.
Message Type(ID)	16	Application message ID The messaging specifications will assign unique Message Type (ID)s.
Message version	8	Application message version The messaging specifications will define the valid versions of each message and its content.
Flags	8	Flags indicating what options were used in constructing the header See paragraph 3.2.1.4, “Flags,” for details.
Data Length	24	Size of message body
Message number	32	Application level message sequence number
Message time	32	Time of message creation 0 if not supported, see paragraph 3.2.2.2, “Message Time Stamp,” for options and details.
Variable header size	8	Size of the variable portion of the header
Time To Live	16	Message time to live (seconds) Used to aid in routing and bridging.
Routing QoS	16	Message Quality of service Used by wireless infrastructure to aid in network selection and prioritization.
Source	64 bytes max	Message source address Null terminated string
Destination	64 bytes max	Message destination address Null terminated string
Body **Message Body**		
Data integrity	32	CRC or application-specific FCS (e.g., HMAC) 0 if not supported

3.2 Message Header Details

3.2.1 Common Header Fields

This section details the fields in the common portion of the EMP header. These fields are required for all EMP messages.

3.2.1.1 Protocol (Header) Version

To allow evolution of the protocol, a header version is specified. Each version of the EMP Message Format Specification will define the applicable header version. Versions in the range of 0x1 to 0xff are valid. 0x0 is not a valid protocol version. Each version of the EMP Message Format specification will contain a section (paragraph 3.4, “Version Numbers”) specifying what version number should be used.

The protocol version may *not* be the same as the EMP Message Format specification version, since revisions to the EMP Message Format specification may not result in changes to the header or behavior.

Note: Previous versions of the EMP Message Format specification defined the protocol version field to be represented in major.minor format (e.g., 1.2). Version 1.0 of the EMP Message Format specification defines the use of protocol version 1.0 (0x8), and versions 1.1–1.3 of the EMP Message Format specification defined the use of protocol version 1.1 (0x9). For backwards compatibility, protocol versions 0x8 and 0x9 shall not be used.

3.2.1.2 Message Type (ID)

The messaging specification will contain the known message types and will assign each of them a unique ID (e.g., GPS Location, Mandatory Directive, and Self-describing). The specifications will also define the contents (data elements and their relative order) of each version of a message type.

Because self-describing messages (see paragraph 3.3.2) can theoretically contain any combination of data fields defined in the data dictionary, message ID 0xffff identifies any self-describing message.

If, for efficiency purposes, a certain domain requires the definition of domain-specific, self-describing messages, this can be accommodated by specifying message types for that domain that are self-describing. An example of this might be a wayside status self-describing message. Such a message would allow flexibility to include any data elements in the message but would also relieve the receiver from having to filter all generic self-describing messages (with id = 0xffff). An alternative to this approach is to include as the first field in a self-describing message a data element that facilitates filtering.

3.2.1.3 Message Version

The messaging specifications will define the valid versions of each type of message.

3.2.1.4 Flags

The Flags field is used to indicate any options the sender exercised while constructing the EMP message (e.g., encryption, time stamp). The following are the definitions for the bits in the Flags field:

3.2.1.4.1 Bit 0: Timestamp format

0 = Relative time (elapsed time since last message creation)
1 = Absolute time (UTC time according to specifications in section 3.2.2.2 Message Timestamp)

3.2.1.4.2 Bit 1: Encryption

0 = Body is not encrypted
1 = Body is encrypted

3.2.1.4.3 Bit 2: Compression

0 = Body is not compressed
1 = Body is compressed

3.2.1.4.4 Bits 3–4: Data integrity

0 = No data integrity support
1 = CRC calculated per specifications in paragraph 3.6.
2 = Application-specific data integrity value used
3 = Reserved

3.2.1.4.5 Bits 5–7: Reserved

Notes:

1. The compression and encryption flags indicate to the message recipient if the message was compressed or encrypted by the sender. This is different from the QoS service indicators requesting that the underlying messaging system compress or encrypt the message.
2. The EMP Message Format specification does not define what specific compression or encryption algorithms are used. Each application (including intermediate nodes if applicable) must agree on the algorithms used.

3.2.1.5 Data Length

The Data Length field specifies how many bytes are in the message body.

3.2.2 Optional Header Fields

This section details the optional fields in the EMP header. Although these fields will always be present in an EMP header, they may not be used by all users and may contain default values.

3.2.2.1 Message Number

Each message can be assigned a unique identifier. The management and interpretation of the message numbers is to be determined by agreement between sender and receiver of the messages.

3.2.2.2 Message Time Stamp

Each message can be time-stamped. If absolute time is used, the meaning of the time stamp (e.g., event time, message creation time, or message transmission time) is not specified as part of the EMP Message Format specification.

The time stamp is optional and can be in either relative or absolute time (see paragraph 3.2.1.4, “Flags”).

The value of the time stamp indicates source support for time-stamping as follows:

- Time stamp = 0: Source device does not support time stamps
- Time stamp != 0: Source devices supports time stamps; interpret time according to the time stamp options flag.

If time stamp is supported (non zero), the options flag indicates the format of the time as follows:

- Relative time: The time stamp indicates the number of seconds elapsed since the last message was created.
- Absolute time: The time stamp indicates the UTC time expressed as the absolute number of seconds since midnight, January 1, 1970, including leap seconds. Using an absolute value implies that times prior to the epoch are not supported (this allows a 32 bit value to not roll over until after the year 2100).

3.2.2.3 Variable Header Size

This field specifies the number of bytes that are used in the variable portion of the header. This field is used by the receiver to help locate the beginning of the message body.

3.2.3 Variable Header Fields

This section details the variable portion of the EMP header. The variable portion of the header may not exist in all EMP messages, but if one of the fields is used, they must all exist (although some of them may be null or empty). The “Variable header size” field indicates the size of this portion of the header.

3.2.3.1 Time to Live

The TTL is the time allocated by the application for an underlying messaging infrastructure to deliver the message to the destination. TTL is an offset from the point in time when the message enters the messaging infrastructure. The allocated, unsigned 16 bits allow a range from 1 second to about 18 hours.

3.2.3.2 Quality of Service (QoS)

The EMP QoS is intended to support the operation of the wireless portion of the messaging infrastructure (messaging system). The messaging system can use the EMP QoS in conjunction with the EMP TTL and EMP message size in support of the following functions:

- Network selection, including the attempt to use different networks at different times within the TTL
- Message prioritization/reprioritization on a selected network
- Providing messaging system services

Note: This is not the IP QoS field.

3.2.3.2.1 QoS Format

Table 3.2, “EMP QoS format,” shows the subfields in the QoS field, and the following sections describe each subfield.

Table 3.2 EMP QoS format

EMP QoS

Bit #	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Service Request			Special Handling				NP			Priority			Class		

3.2.3.2.2 Class

EMP messages are assigned one of eight message classes[a/]. The message class is used during route selection. Each end of a transport network is configured with a list of message classes that can be transmitted over that transport network. Messages whose assigned class is a member of the list of classes allowed over a transport network are eligible for transmission using the transport network[b/]. This attribute represents any constraints on usage of a transport network such as actual dollar cost or bandwidth. Message Class does not specify what transport networks should be used but can preclude a transport network from usage.

In an interoperable scenario, Message Class is expected to be negotiated between network nodes. Also, how QoS is used must be consistent. Therefore, an interoperable messaging system must provide an unambiguous definition to application developers as to how a message in a given class will be processed by an underlying transport.

3.2.3.2.3 Priority

EMP messages are also assigned one of eight priority levels that are used to indicate to the underlying messaging system the priority of a message relative to all other messages in the system. The messaging system must provide an unambiguous definition of how message priority is consistently used (e.g., fair-share processing between priorities and various scheduling algorithms within a priority).

3.2.3.2.4 Network Preference

An EMP message may indicate a transport network preference that is used to override the default network selection behavior of the messaging system. The application does not explicitly specify a transport network in the attribute; instead, the application uses a value that is mapped (via configuration) to a specific network by the messaging system. The Transport Preference can indicate one of the following three preferences:

- No preference: use default routing behavior.
- All transports: transmit the message on all available transports that have a path to the destination simultaneously.
- Specific transport: If a path exists to the destination or becomes available within the TTL using the preferred transport, use that transport even though other paths may exist over other transports. The messaging system will use the preferred transport even though other transports would have been deemed preferable using the default route-selection logic.

[a/] The names of the EMP classes, guidelines for assigning a class to a message, and the class assignments of existing messages (e.g., ITC) will be defined the by the AAR and ITC at a future date.

[b/] This describes a linear implementation of message Class. See description of non-linear implementation described later in this same section

Using a network preference is discouraged and should be used only in situations where allowing the messaging system to select the network based on QoS class and priority is known to be insufficient (e.g., ITC wayside 220 preference). The following are the defined values for the Network Preference field:

0 (000)	=	No preference
1 (001)	=	Defined by the messaging system (220 MHZ for ITC/ITCM)
2 (010)	=	Defined by the messaging system
3 (011)	=	Defined by the messaging system
4 (100)	=	Defined by the messaging system
5 (101)	=	Defined by the messaging system
6 (110)	=	Defined by the messaging system
7 (111)	=	Send message on all available networks simultaneously

3.2.3.2.5 Special Handling

An EMP message may include special handling instructions that are used by the messaging system to perform non-standard processing[a/] for that message. The application does not explicitly specify what the special handling is. Similar to the Network Preference attribute, the application instead specifies a value that is mapped by the messaging system (via configuration) to one or more special handling steps. The special handling code may be passed through various layers of the messaging stack, and each may act on it if configured to do so. The following are the defined values for the Special Handling field:

0 (0000)	=	No special handling required
1–31 (0001 - 1111)	=	Reserved

3.2.3.2.6 Service Requests

This field is used to indicate to the underlying messaging system the services desired by the application. Not all services defined in the EMP Service Requests field may be supported by all messaging systems. If the application requests a service that is not supported, the messaging system shall return an error indication to the application in an EMP outcome message.

Table 3.3 EMP QoS service requests

EMP QoS Service Requests Field			
EMP QoS bit #	15	14	13
Service Requests subfield bit #	2	1	0
	Compression requested	Delivery acknowledgement	Outcome notification
	0: No message compression requested	0: No end-to-end delivery acknowledgement requested	0: No notification requested
	1: Message compression requested	1: End-to-end delivery acknowledgment requested	1: Outcome notification requested

[a/] Examples of special handling include the following:
~Stripping and reconstitution of message headers
~Message header modification (e.g., replacement of destination addresses)
~Special handling at the RF level, e.g., use of CSMA vs. TDMA, use of certain RF channels or message types.

3.2.3.2.6.1 Outcome Notification

If outcome notification is requested and supported, the messaging system shall asynchronously send to the application an EMP message indicating the outcome of the transmission attempt. Note that even if an application does *not* request an outcome notification, it may still receive such a message for error reporting.

3.2.3.2.6.2 Delivery Acknowledgement

If delivery acknowledgement is requested and supported, the messaging system shall implement end-to-end delivery acknowledgement for the specified messages. Message delivery is defined as the messaging system having delivered the message to the application (does not indicate any information about the application accepting or processing the message). The delivery acknowledgement will be communicated to the application using an EMP Outcome Notification message.

3.2.3.2.6.3 Compression

If compression is requested and supported, the messaging system shall compress the message body prior to transmission. The message shall be decompressed by the messaging system prior to delivery to the destination specified in the EMP destination field. An out-of-band protocol is used to ensure that messaging system components are using compatible compression algorithms. Note that making a compression request (setting bit 15 in the EMP QoS) is different from the application setting the compressed bit in the Flags field of the EMP header (paragraph 3.2.1.4, “Flags”) indicating that the application has compressed the message itself.

3.2.3.3 Source and Destination

EMP is intended to wrap industry interoperable messages that will travel over interoperable network transports (wired and wireless) and is expected to be bridged into and out of the various messaging infrastructures at various organizations and on their various assets. The EMP source and destination addresses are logical, network-, and infrastructure-independent names. Although in that context a format specification is not required, please see Appendix A, “Requirements for Using an ITC EMP Application Gateway,” for a specification of the format to be used when messages are being transported through the ITC Messaging System.

3.2.3.3.1 Source

This field identifies the source of the message as a null-terminated string with a limit of 64 bytes. It is important to note that although this can be a DNS-compatible name, the use of DNS is not required. An empty source address (e.g., only the null terminator) is a valid source address.

3.2.3.3.2 Destination

This field identifies the final destination of the message as a null-terminated string with a limit of 64 bytes. It is important to note that although this can be a DNS-compatible name, the use of DNS is not required. An empty destination address (e.g., only the null terminator) is a valid destination address.

3.2.4 Data Integrity

This field is used to support data integrity. This field can also support message authentication if an appropriate application-specific DIV value is used (e.g., HMAC). The contents of the field will vary depending on the value in the Flags field (see paragraph 3.2.1.4, “Flags”). If a CRC is used in this field, it will contain a 32 bit CRC calculated according to the specifications in paragraph 3.6, “CRC Calculation.” If a data integrity value is not supported by the sender, this field will have a value of 0.

If a DIV value is used (either CRC or application-specific), the DIV shall cover the entire EMP message (all header fields and message body).

3.3 Message Formats

The contents and format of the EMP message body are application-specific. But in general, at least two types of message formats will be supported: fixed format and self-describing. This section discusses each format in more detail.

3.3.1 Fixed-Format Messages

The format of fixed-format messages will be determined by a messaging specification that will specify the following:

- The ID of the message
- The data elements in the message (and their data dictionary element ID)
- The order of the data elements in the message

Fig. 3.1 illustrates how a fixed-format message would be structured.

Header
EMP Header
Body
Data Element #1
Data Element #2
....
Data Element #n
End of Body
Data integrity

Fig. 3.1 Fixed format message

3.3.2 Self-Describing Messages

Self-describing messages can contain any number and combination of data elements defined in the data dictionary. Self-describing messages are the most flexible kind of message, but are also the most expensive to construct, transmit, and parse. They are a very restricted form of binary XML. Each data element in the self-describing message is preceded by a tag that contains the data element ID as defined in the data dictionary. The data dictionary contains the definition of format in which the data element will be stored in the message. Following each data tag is the size of the data elements in bytes. This field allows applications that are unable to process the field to find the next field.

Self-describing messages will have a fixed-message type assigned to them (Message Type (ID) = 0xFFFF). Fig. 3.2, “Self-describing message example,” illustrates how a self-describing message would be structured.

Header
EMP Header
Body
Data element #1 ID
Data element #1 size (bytes)
Data element #1 value
Data element #2 ID
Data element #2 size
Data element #2 value
………………….
Data element #n ID
Data element #n size
Data element #n value
End of Message marker
End of Body
Data integrity

Fig. 3.2 Self-describing message example

3.4 Version Numbers

The following version number shall be used as of this version of the EMP Message Format specification.

Table 3.1 Version number definitions

Field	Version
EMP header version	0x4(4)

3.5 EMP Protocol Messages

Although EMP is primarily a message envelope, some of the fields in the envelope imply certain behavior from the sender or receiver of a message. The behavior may entail sending or replying with a certain EMP message. Examples of such messages are an outcome notification message or a subscription message.

To allow for variances in messaging system capabilities, this specification will not define the contents of such messages but will define the minimum information that should be in such messages and any header fields that are expected to be common across all messaging systems.

3.5.1 Outcome Notification Message

The Outcome Notification Message is used by a messaging system to provide asynchronous feedback to the application regarding messages submitted for transmission. The exact format and contents of the Outcome Notification Message will be defined by each messaging system, but at a minimum will contain the following:

- Reference message ID—the EMP sequence number of the message that was submitted to the underlying transport and to which this outcome pertains
- Outcome code—an indication of the outcome of the message transmission. Candidate outcome codes are
  - TTL expiration
  - End-to-end delivery confirmation
  - Error reports (e.g., ill-formatted header fields)

The EMP header of any outcome notification message shall contain the following field settings:

- TDB

3.5.2 Subscribe Message

3.5.3 Unsubscribe Message

3.5.4 Subscription Confirmation Message

3.6 CRC Calculation

Following is the specification for the CRC-32 algorithm that is used in PKZip, AUTODIN II, Ethernet, and FDDI.

Name:	“CRC-32”
Width:	32
Poly:	04C11DB7
Init:	FFFFFFFF
RefIn	True
RefOut::	True
XorOut:	FFFFFFFF
Check:	CBF43926

Table-driven implementation:

```
#define   B   8
static unsigned long crcTable[1<<B]={
0x00000000L,0x77073096L,0xEE0E612CL,0x990951BAL,0x076DC419L,0x706AF48FL,0xE963A535L,0x9E6495A
3L,
0x0EDB8832L,0x79DCB8A4L,0xE0D5E91EL,0x97D2D988L,0x09B64C2BL,0x7EB17CBDL,0xE7B82D07L,0x90BF1D9
1L,
0x1DB71064L,0x6AB020F2L,0xF3B97148L,0x84BE41DEL,0x1ADAD47DL,0x6DDDE4EBL,0xF4D4B551L,0x83D385C
7L,
0x136C9856L,0x646BA8C0L,0xFD62F97AL,0x8A65C9ECL,0x14015C4FL,0x63066CD9L,0xFA0F3D63L,0x8D080DF
5L,
0x3B6E20C8L,0x4C69105EL,0xD56041E4L,0xA2677172L,0x3C03E4D1L,0x4B04D447L,0xD20D85FDL,0xA50AB56
BL,
0x35B5A8FAL,0x42B2986CL,0xDBBBC9D6L,0xACBCF940L,0x32D86CE3L,0x45DF5C75L,0xDCD60DCFL,0xABD13D5
9L,
0x26D930ACL,0x51DE003AL,0xC8D75180L,0xBFD06116L,0x21B4F4B5L,0x56B3C423L,0xCFBA9599L,0xB8BDA50
FL,
0x2802B89EL,0x5F058808L,0xC60CD9B2L,0xB10BE924L,0x2F6F7C87L,0x58684C11L,0xC1611DABL,0xB6662D3
DL,
0x76DC4190L,0x01DB7106L,0x98D220BCL,0xEFD5102AL,0x71B18589L,0x06B6B51FL,0x9FBFE4A5L,0xE8B8D43
3L,
0x7807C9A2L,0x0F00F934L,0x9609A88EL,0xE10E9818L,0x7F6A0DBBL,0x086D3D2DL,0x91646C97L,0xE6635C0
1L,
0x6B6B51F4L,0x1C6C6162L,0x856530D8L,0xF262004EL,0x6C0695EDL,0x1B01A57BL,0x8208F4C1L,0xF50FC45
7L,
0x65B0D9C6L,0x12B7E950L,0x8BBEB8EAL,0xFCB9887CL,0x62DD1DDFL,0x15DA2D49L,0x8CD37CF3L,0xFBD44C6
5L,
0x4DB26158L,0x3AB551CEL,0xA3BC0074L,0xD4BB30E2L,0x4ADFA541L,0x3DD895D7L,0xA4D1C46DL,0xD3D6F4F
BL,
0x4369E96AL,0x346ED9FCL,0xAD678846L,0xDA60B8D0L,0x44042D73L,0x33031DE5L,0xAA0A4C5FL,0xDD0D7CC
9L,
0x5005713CL,0x270241AAL,0xBE0B1010L,0xC90C2086L,0x5768B525L,0x206F85B3L,0xB966D409L,0xCE61E49
FL,
0x5EDEF90EL,0x29D9C998L,0xB0D09822L,0xC7D7A8B4L,0x59B33D17L,0x2EB40D81L,0xB7BD5C3BL,0xC0BA6CA
DL,
0xEDB88320L,0x9ABFB3B6L,0x03B6E20CL,0x74B1D29AL,0xEAD54739L,0x9DD277AFL,0x04DB2615L,0x73DC168
3L,
0xE3630B12L,0x94643B84L,0x0D6D6A3EL,0x7A6A5AA8L,0xE40ECF0BL,0x9309FF9DL,0x0A00AE27L,0x7D079EB
1L,
0xF00F9344L,0x8708A3D2L,0x1E01F268L,0x6906C2FEL,0xF762575DL,0x806567CBL,0x196C3671L,0x6E6B06E
7L,
0xFED41B76L,0x89D32BE0L,0x10DA7A5AL,0x67DD4ACCL,0xF9B9DF6FL,0x8EBEEFF9L,0x17B7BE43L,0x60B08ED
5L,
0xD6D6A3E8L,0xA1D1937EL,0x38D8C2C4L,0x4FDFF252L,0xD1BB67F1L,0xA6BC5767L,0x3FB506DDL,0x48B2364
BL,
0xD80D2BDAL,0xAF0A1B4CL,0x36034AF6L,0x41047A60L,0xDF60EFC3L,0xA867DF55L,0x316E8EEFL,0x4669BE7
9L,
0xCB61B38CL,0xBC66831AL,0x256FD2A0L,0x5268E236L,0xCC0C7795L,0xBB0B4703L,0x220216B9L,0x5505262
FL,
0xC5BA3BBEL,0xB2BD0B28L,0x2BB45A92L,0x5CB36A04L,0xC2D7FFA7L,0xB5D0CF31L,0x2CD99E8BL,0x5BDEAE1
DL,
0x9B64C2B0L,0xEC63F226L,0x756AA39CL,0x026D930AL,0x9C0906A9L,0xEB0E363FL,0x72076785L,0x0500571
3L,
0x95BF4A82L,0xE2B87A14L,0x7BB12BAEL,0x0CB61B38L,0x92D28E9BL,0xE5D5BE0DL,0x7CDCEFB7L,0x0BDBDF2
1L,
0x86D3D2D4L,0xF1D4E242L,0x68DDB3F8L,0x1FDA836EL,0x81BE16CDL,0xF6B9265BL,0x6FB077E1L,0x18B7477
7L,
0x88085AE6L,0xFF0F6A70L,0x66063BCAL,0x11010B5CL,0x8F659EFFL,0xF862AE69L,0x616BFFD3L,0x166CCF4
5L,
0xA00AE278L,0xD70DD2EEL,0x4E048354L,0x3903B3C2L,0xA7672661L,0xD06016F7L,0x4969474DL,0x3E6E77D
BL,
0xAED16A4AL,0xD9D65ADCL,0x40DF0B66L,0x37D83BF0L,0xA9BCAE53L,0xDEBB9EC5L,0x47B2CF7FL,0x30B5FFE
9L,
0xBDBDF21CL,0xCABAC28AL,0x53B39330L,0x24B4A3A6L,0xBAD03605L,0xCDD70693L,0x54DE5729L,0x23D967B
FL,
0xB3667A2EL,0xC4614AB8L,0x5D681B02L,0x2A6F2B94L,0xB40BBE37L,0xC30C8EA1L,0x5A05DF1BL,0x2D02EF8
DL
};

unsigned long CCRC32::crcCalculation(unsigned char* buffer,int bufferLength)
{
    unsigned long crc=0xFFFFFFFFL;
    while(bufferLength--)
        crc=crcTable[(crc^*buffer++)&0xffL]^(crc>>B);
    return(crc^0xffffffff);
}
```

When compiled and run against the ASCII string “123456789,” the table-driven implementation yields the check value from the specification, 0xCBF43926.

Note: To validate the checksum transmitted with a message, recalculate the checksum over the message (sans checksum) and compare the result with the transmitted checksum. Optionally, complement the transmitted checksum (XOR the transmitted checksum with 0xFFFFFFFF), calculate the checksum over the message and the transmitted checksum, complement the result, and compare with 0x00000000.

Nine-byte check message with four-byte (big-endian) CRC:

0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0xcb 0xf4 0x39 0x26

The checksum calculated over the first nine bytes will match the last four bytes.

Nine-byte check message after complementing the transmitted checksum:

0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0xd9 0xc6 0x0b 0x34

the CRC calculated over the entire thirteen bytes will be 0xFFFFFFFF; its complement is, of course, 0x00000000.

4.0 PROTOCOL SECURITY REQUIREMENTS

No security mechanisms or behaviors are defined by EMP.

5.0 PROTOCOL PERFORMANCE REQUIREMENTS

No performance requirements are specified in this specification.

APPENDIX A
REQUIREMENTS FOR USING AN ITC EMP APPLICATION GATEWAY

1.0 SCOPE

Interoperable Train Control (ITC) application will use EMP as the application message envelope and will use the ITC Messaging System to transport the messages between applications. The entry point into the ITC Messaging system is an ITC EMP Application Gateway (AG). The function of the AG is to act as a bridge between the application protocols (EMP) and the Messaging System Protocols (ITCM). To facilitate this bridging function, the ITC EMP AG requires that certain EMP fields be used according to specifications in this section.

2.0 SOURCE AND DESTINATION ADDRESS

The EMP source and destination addresses must follow the grammar specified in this section. The EMP address grammar is written in Augmented Backus-Naur Form[a/,b/]. The grammar has just a few symbols for communicating the production rules:

<Rule name> ::= rule text	the format of a production rule
(...)	encloses a expression
[ ]	encloses an optional expression (i.e., expressions that are not optional are required)
n*m	defines the number of times an expression can repeat with minimum n and maximum m and refers to the expression that immediately follows it
*m	a repetition from a minimum of zero to a maximum of m
|	means logical “or” (i.e., if expressions are not separated by “|” then “and” is assumed)
n-m	defines a range of values

2.1 Format

```
1.  <EMP Address> ::= null | (<network name> ": " <messaging name>)
2.  <network name> ::= <owning organization>"."<asset>
3.  <owning organization > ::= 2*4(alpha)
4.  <asset> ::= <locomotive> | <wayside> | <back office> | <virtual remote>
5.  <locomotive> ::= "l."<locomotive identifier>
6.  <locomotive identifier> ::=<initials>"."<asset number>
7.  <initials> ::= 1*4(alpha)
8.  <asset number> ::= 1*6(number)
9.  <wayside>::= "w."<wayside identifier>
10. <wayside identifier>::=6(number)
11. <back office>::="b"
12. <virtual remote>::="v".<virtual remote identifier>
13. <virtual remote identifier>::=<asset number>
14. <messaging name> ::= <string> ["." <string>]
15. <string> ::= *(<alpha> | <number>)
16. <alpha> ::= a-z | A-Z
17. <number> ::= 0-9
```

[a/] Augmented BNF for Syntax Specifications: ABNF, RFC 2234
[b/] Routing Backus-Naur Form (RBNF): A Syntax Used to Form Encoding Rules in Various Routing Protocol Specifications, RFC 5511

The maximum allowable length of the produced address is 64 characters. As an example, the shortest network name that could be produced is 4 (i.e., "ns.b"), so the longest acceptable messaging name is 58 characters (reserving 2 characters for the separator ":" and null terminator).

All character strings are case insensitive.

2.1.1 Owning Organization

The owning organization portion of the source/destination address is a string that uniquely identifies an organization within the industry. These strings shall be defined in the AAR Messaging Dictionary. Some examples of owning organization names are "bnsf," "csx," "ns," "up," "rlnc," and "aar."

2.1.2 Asset

The <asset > portion of the source/destination address defines the kind of entity to which the address is referring. In the current version, this is limited to Locomotive, Wayside, Back Office, or virtual remote. Other assets are possible, and their definitions can be readily added to the grammar.

2.1.3 Locomotives

A Locomotive consists of a single character "l" followed by a locomotive identifier. The identifier in turn consists of up to four alpha initials followed by one to six numeric digits.

2.1.4 Wayside

A Wayside consists of a single character "w" followed by a wayside identifier. The identifier in turn consists of six numeric digits. For ITC applications, the LLL.GGG portion of the ATCS address assigned to the wayside will be used.

2.1.5 Back Office

A Back Office consists of a single character "b." Back office assets do not require a back office identifier.

2.1.6 Virtual Remote

A Virtual Remote consists of a single character "v" followed by a virtual remote identifier. The identifier in turn consists of one to six numeric digits.

2.1.7 Messaging Name

The <messaging name> portion of the EMP address is intended to aid moving the message through the messaging infrastructures in the various back offices and on various remote assets. It is a dot-separated string used by EMP message routers.

2.2 EMP Address Examples

Examples:

```
up.l.5560:rumpelstiltskin
up.l.sp.123:switcher.locations
ns.l.hclx.936012:itc.vtms
cccl.l.c.1:itc
up.b:itc.bos1
bnsf.b:ptc
csx.b:cbtm
up.b:events.optitrac.zoneEvents.ABT
ns.w.123456:78
ns.w.123456:78.sm
csx.v.1234:TMC
```

APPENDIX B
APPLICATION NOTES (EMP NODE EXAMPLES)

1.0 SCOPE

A typical infrastructure will include various numbers and types of EMP nodes. An EMP node is any process that accepts or generates EMP messages over any message transport protocol. A typical infrastructure will include terminal nodes such an application and may also include relay type nodes such as EMP message routers, bridges, and monitors. These functions are logical and may be packaged together in a single physical process or application.

2.0 EMP TERMINAL NODE

An EMP terminal node is a process that has one or more message transport links carrying EMP message traffic. All message traffic into a terminal node terminates in the node (e.g., processed by the node). Similarly, all message traffic out of a terminal node is generated by the node. Applications and bridges are examples of terminal nodes.

3.0 EMP MESSAGE ROUTER

An EMP message router is a process that has one or more (at least two to be useful) message transport links carrying EMP message traffic. The message router's job is to route incoming messages out to zero or more of its links. The routing is typically done based on evaluating the EMP header against a set of rules. The routing algorithm and capability are dependent on the specific router and are not part of the specifications. Similarly, how the router gets programmed can vary widely from static configuration to dynamic subscriptions via EMP messages. Again, the method of specifying the routing knowledge will be a function of the specific router. A draft proposal is included for the EMP messages that would enable message-based router configuration.

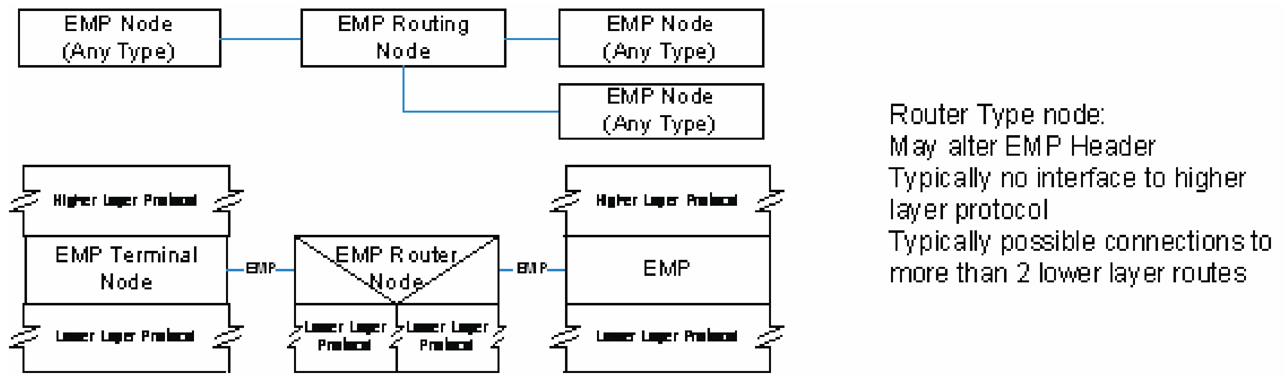

Fig. B.1 EMP router example and stack

4.0 EMP MESSAGE GATEWAY

An EMP message gateway is a process at integration boundaries (edge) between messaging technologies. A gateway allows an application using EMP to communicate with another EMP application through a different messaging system (e.g., JMS, ITCM). A gateway also allows an EMP application to communicate with an application using a different messaging system. An EMP gateway has one or more message transport links carrying EMP message traffic on one side and some other message transport carrying a different message format (e.g., JMS, MQ Series, MSC3) on the other side. The gateway's job is to generate zero or more outgoing messages in the other messaging system for each incoming EMP message and to generate zero or more outgoing EMP messages for each incoming message from the other messaging systems. What messages are generated and the algorithm(s) for generating those messages will vary between gateway implementations and the messaging system being used. EMP messaging gateways (e.g., EMP to JMS) operate based on information in the EMP header. Application-specific messaging gateways (e.g., PTC-EMP to PTC-JMS) may operate based on both the EMP header and the contents of the EMP messages.

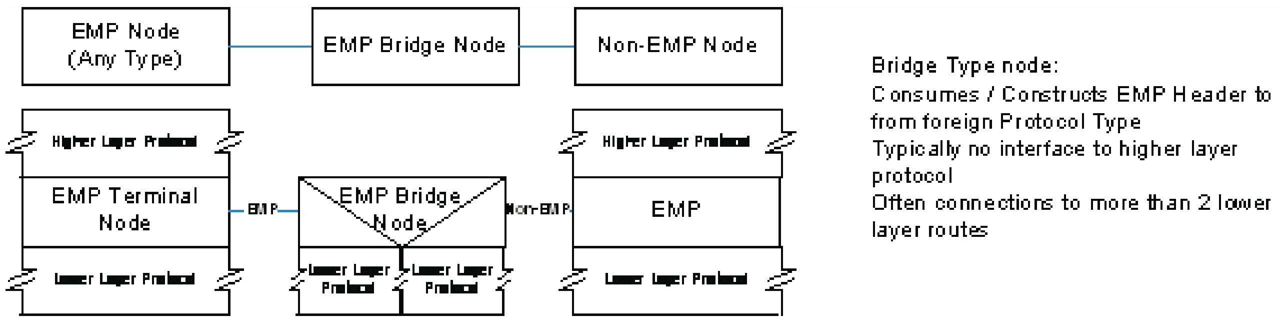

Fig. B.2 EMP message bridge example and stack

5.0 EMP MESSAGE MONITOR

An EMP message monitor is a process through which EMP message traffic flows and is monitored. The most common type of EMP monitor is a time monitor that monitors for excessive delays between arriving message time stamps and receiving node time stamps. This is typically used in a High Availability environment. There are many other types of attributes that can be monitored, such as message types, source, destinations, size, number, etc.

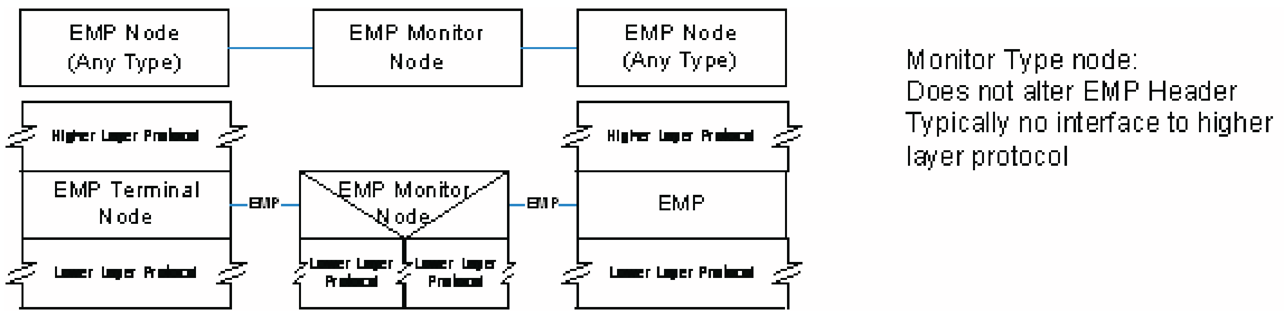

Fig. B.3 EMP message monitor example and stack