Development of a Wayside Detector Open Communication Standard

HAROLD HARRISON

There is an emerging trend in the railroad industry to consolidate more wayside detectors at fewer installations. By combining several detectors and auxiliary devices, such as vehicle identification equipment (cameras and tag readers), considerable efficiencies are achievable, which, in turn, benefit the growth toward automatic train control systems (ATCS) integration. Given that the various devices are not generally available from a sole supplier, a standardized means of communicating among all devices is obviously needed. In building a framework for the development of such a standard, the primary goal is to separate information into relative groups by the nature of their respective time criticality, the quantity of information passed, and the relative capacity of each device involved to handle its task.

Most microprocessor-based detectors have at least some limited communication capacity via a serial port (usually RS-232–D, 8 bits, 1 stop, no parity). Because ASCII code is the most common form of information sent over serial communication lines, it makes sense to use ASCII as the basis for any message (whether command or data). ASCII characters also provide unique bit patterns that can be reserved for specific functions. Development and debugging are simple as well, using text editors and terminal emulators on personal computers. There are, however, fundamental limitations to this approach: the finite time necessary to transmit a message, and having sufficient "smarts" in any one particular device to communicate in this mode and to do it in a real-time manner.

There are several different criteria regarding timing and time delays or uncertainties that have some significance in defining "real time" as it applies to the wayside environment. For instance, for many activities, 1 msec is about the limit of "instantaneous response." This is equivalent to about 1 in. (25 mm) of uncertainty of position for train speeds of 60 mph (100 km/hr). One msec is probably the lower limit of timing control of separate devices and is coincidently the time it takes to transmit one ASCII character at 9600 baud. This arbitrary definition could be used as a starting point for several classes of activity:

1. Submillisecond timing—Presently this is beyond the scope of the proposed specification, except for dedicated control lines that could be set up between devices. (Salient has accommodated activity between its detector subsystems by using high-speed multidrop serial links and distributed reference timers that are synchronized to about 100μ sec.)
2. Millisecond timing—This is the practical lower limit of this basic approach using single ASCII character messages. Because addressing is not possible under these constraints, this mode would be restricted to one device on each end of serial link.
3. Near real time (multimillisecond)—Sufficient time is available to transfer more complete commands and small data messages. This timing uncertainty should be sufficiently tight to never misidentify specific axles. This mode is still too time restrictive to allow multidrop communication mode or receiver acknowledge.
4. Negative time—This class allows for events that cannot be handled in real time. For example, one detector may be located down the track so that its "real time" is much later than that of the device to which it talks. Obviously, this requires that the equipment receiving the message have buffering that accommodates the longest negative time specified.
5. Future time—Provision for preloading messages that apply at some point in the (near) future specified to 1 msec. This would allow two real-time devices to handle this communication at lower priority than that needed for immediate response. It further implies clock synchronization between the two devices, unless time is relative to transmission.
6. Posttrain activity—This is the worst-case condition of negative time that applies to most current detectors that may have only a posttrain mode of communicating over a serial link. In addition, many devices may pass final judgment on their respective alarms after reviewing their complete records. This indicates an important need for a cleanup phase of operation that may undo prior activity, as well as accommodate longer messages that would bog down the communication process during the time-critical period.

MESSAGE FORMATS

Borrowing from microcomputer instruction techniques, different classes of messages can be defined:

1. "Immediate" messages—In keeping with Type 2 timing above, there should be provision for single ASCII character messages. These should be generic in nature so that each detector can specify the exact response it expects. For reasons explained below, these single character instructions will be restricted to lowercase alpha, excluding "a,b,c,d,e,f". This provides 20 total immediate instructions (which can be reassigned for different interfaces).
2. Normal messages—These messages are four or six ASCII characters in length, and the last character is a checksum. These messages would be available for the near real-time activities, with no acknowledge expected. As detailed

Salient Systems, Inc., 4140 Tuller Road, Dublin, Ohio 43017.

below, these begin with an uppercase alpha followed by two or four data characters that are tentatively restricted to decimal or hex values (using lowercase "a" through "f" for upper hex numbers). Two leading opcode characters could be allowed at the expense of maximum data value.

3. "Extended" messages—This category allows a verbose exchange of information during the posttrain period using either begin and end braces "{,}" (printable ASCII) or "STX,ETX" (unprintable ASCII) to enclose the message string. The internal format of the string can be flexible depending on the mutual agreement of the parties involved. Ordinarily, it should have a header defining the type of message (and possibly message length), a body allowing most printable characters, and a tail with provision for a checksum.

The main purpose in configuring each type of message differently is to allow a "smart" device driver servicing the serial port to determine the type of action needed by interpreting the lead character of each message. In particular, a leading lowercase alpha could cause an immediate interrupt that, for instance, may not be desirable for an extended message.

DETAILED DISCUSSION

Real-Time/Immediate Messages

The greatest danger of applying the single-character messages to achieve the 1-msec response time lies in the compromised reliability. Not only is there no verification mechanism, but many times serial ports may behave less reliably on the transmission of a first or single character (due to flushing of buffers, etc.). The key issue is the trade-off between the actual reliability between two specific devices and the alternative of incorporating another interface that can meet the 1-msec timing limit. Increasing the RS232 baud rates to 19.2 or 38.4 kbaud would certainly be one possibility given that the detection devices are probably in the same or adjacent racks in the wayside enclosure. This would allow moving up from the "immediate" to the "normal" message type while still staying within the 1-msec response domain.

Another approach to meet the 1-msec criterion is to incorporate a faster interface. Many single-chip micros have high-speed ports for local (on-board) connections that can be interfaced to other standard serial interfaces such as RS-422, RS-423, and RS-485. These standards allow for an order of magnitude or more of a speed increase. Obviously, this approach works only if both devices are able to incorporate the higher-speed port.

Near Real-Time/Normal Messages

As summarized above, these messages would be four or six ASCII characters in length, with the last character being a checksum, and no acknowledge expected. Normal messages begin with an uppercase alpha followed by two or four data characters that are tentatively restricted to decimal or hex values (using lowercase "a" through "f" for upper hex numbers). There are plenty of opportunities for subsets in this category that implicitly use decimal or hex exclusively, as well as being implicitly four or six characters in length. Two leading opcode characters could be allowed at the expense of maximum data value.

The details of this message format are not particularly important up to the point that they become a de facto standard after someone has implemented them. The main point is that with a little care there can be some inherent redundancy in the message structure that would improve its reliability without overly taxing the link.

Some devices, such as automatic equipment identification (AEI) tag readers, operate within this area of near real time, currently without any benefit of standardization other than their own. Generally, this means that whatever device receives the tag reader's message string also must recognize the format and make its own assumption about the timing of the message arrival and the corresponding moment when the tag was correctly read. This happens to work reasonably well because there is a fair amount of uncertainty as to when the tag gets read within the field of its antenna. This is also a case of a relatively long ASCII string that must be treated as a near real-time event.

Posttrain/Extended Messages

This area of the proposed standard has the most flexibility, and coincidently it also has the most work already invested in it. Because most detection devices are ordinarily designed to stand alone, it is understandably easier for most devices to handle their own real-time jobs and then transfer the results to other devices after train passage.

Because of the added time available, the extended messages can afford to be (and are recommended to be) acknowledged by the receiving device. ASCII "ACK" and "NAK" characters are proposed. Additionally, busy and error conditions should be returned when appropriate.

The means of properly correlating events during train passage are usually related to time since train arrival (signaled between designated sensor device and all others needing the sync signal) and/or to the axle number associated with a particular event. Some devices can also sense car count or car position in the train, but the lower reliability of this capability suggests the added redundancy of time and axle count is desirable. Consistent with the rest of the standard, the reported time resolution is generally to 1 msec.

Abbreviated Form of Extended Message

For devices with limited ability to create elaborate messages because of code and data space limitations, a simplified version of the extended message is proposed. The message string would begin with two uppercase alpha opcode characters after the beginning brace or "STX". Any number of following data characters (up to some practical limit such as 72) would be followed by an end brace or "ETX" and two hex checksum characters. The data field can include any uppercase alphas for text and hex or decimal numbers. Space, tab, and basic punctuation characters are allowed for separation and visual convenience. In this case, the checksum is suggested to occur after the end brace. This allows simpler devices to strip the

checksum off without much difficulty if they are incapable of using the integrity check.

Given the limited capability of the devices that might use this version of the extended message, it is likely that message acknowledgement would not be incorporated here.

Note that regarding the choice between braces and STX, ETX characters, braces have the advantage of being printed to a terminal screen, thus allowing for ease of development and debugging with simple techniques.

One of the fringe benefits of the proposed standard is the potential for separate contractors to exchange simple text files in order to simulate each other's equipment.

Fully Developed Form of Extended Message

The appendix presents a more complete form of extended message. This message structure is consistent with the abilities of larger detection devices that have some reserve capacity to handle more complete communication sessions. The details of this message format represent the fruits of much effort on the part of the Video Masters staff after numerous sessions with Salient Systems' crew.

CONCLUSIONS

There is a demonstrated need for a set of ground rules to cover communications between various wayside detection devices. This paper is an attempt to define some of the inherent limitations that are present in the wayside detection process and with the current equipment available to perform these tasks. A set of guidelines were then outlined and the framework for a communication standard presented. It is hoped that this effort will provide an easy starting point from which to adopt a formal standard for acceptance by the railroad industry as a whole.

ACKNOWLEDGMENTS

The author wishes to thank the Burlington Northern and Union Pacific Railroads for providing the encouragement to resolve this issue. In addition, the staff at Video Masters, Servo, and Salient Systems have put in many long hours to hammer out many of the details and start putting them to use in the field.

APPENDIX
Extended Message Formats (Version 1)

General Format: [header] { message stream } [tail]

All data are expressed in ASCII. All transmitted hexadecimal data are the ASCII representations of the hexidecimal values.

Header Format: [header] = [STX] [version] [nchar]

field name	[STX]
size (char)	1
type	constant
description	02H
field name	[version]
size (char)	1
type	character
description	Provides interface version number for future expansion and compatibility, starting with "1".
field name	[nchar]
size (char)	4
type	hexchar
description	Total characters in message stream (hex count)
special values	**ffff** = test data (of undetermined length) in message, look for special tail indication for end of data.

Tail Format: [Tail] = [cksum switch] [cksum] [ETX]

special values **ffff,ETX** indicate end of data for text files.

field name	[cksum switch]
size (char)	2
type	hexchar
description	indicates that cksum field is active
special value	**ff** = cksum *does not* have to be checked.
field name	[cksum]
size (char)	2
type	hexchar
description	2's complement of the 8-bit sum of all the binary equivalents of ASCII characters in message stream
special value	**ff** when combined with **ff** above closes the undetermined length, text file declared in header.
field name	[ETX]
size (char)	1
type	constant
description	03H

General format of message streams:
{ message stream } = [mtype]{ message fields }

Field Descriptions found among different message types:

field name	[mtype]
size (char)	1
type	upper case alpha
description	Identifies message type, establishes record structure.
field name	[d]
size (char)	1
type	char
description	Delimiter to separate data within a record (e.g. comma, dash, space, or tab).
field name	[d2]
size (char)	2
type	char
description	Delimiter to separate data within a record (normally CR/LF characters).
type description	timestamp [MSB] [ ] [ ] [ ] [LSB] Data are sent in order, from MSB to

LSB. All values are the ASCII representations of the hexidecimal. Assume leading zero's to pad message.

Message 'T': Train Data

The train data consists of the following fields: [mtype][d2][date/time][d][TrnOD][dir][d]-[naxles][d] [alarms][selft][d][TotTime]

field name	[mtype]
size (char)	1
type	constant
description	'T'
field name	[date/time]
size (char)	10
type	decimal
description	date and time train arrived at site (24 hr. format)
field name	[TrnOD]
size (char)	2
type	hexchar
description	(train of day) train number after midnight
field name	[dir]
size (char)	1
type	char
description	1 = left to right; 0 = right to left.
field name	[naxles]
size (char)	3
type	hexchar
description	total number of axles in train
field name	[alarms]
size (char)	2
type	hexchar
description	alarm count (' ' = no alarms)
field name	[selft]
size (char)	2
type	char
description	2 characters indicating system status
field name	[TotTime]
size (char)	6
type	timestamp (hexidecimal)
description	total time (in msecs) TRAIN PRESENT signal was engaged for train.

Message 'A': Axle Data

The axle data consist of the following fields: [mtype] { axle records }

field name	[mtype]
size (char)	1
type	constant
description	'A'
record name	axle record
size (char)	12 per record, up to 1023 records
structure	[d2] [axle number] [d] [axle timestamp]
description	provides the timestamp for axle events
field name	[axle number]
size (char)	3
type	hexchar
description	assigns a number value to an axle event
field name	[axle timestamp]
size (char)	6
type	timestamp (hexidecimal)
description	Time of axle event since TRAIN PRESENT signal was engaged.

Message 'E': Event (or Exception) Data

The event data consist of the following fields: [mtype] { Event records }

field name	[mtype]
size (char)	1
type	constant
description	'E'
record name	Event record
structure	[d2][Event axle number][d][Event type][d][Event label][d][event time]
size (char)	31 per record, up to approx. 100 records
description	Provides a description of significant events (typically, an alarming axle).
field name	[Event axle number]
size (char)	3
type	hexchar
description	Assigns an axle number value to the event or alarm.
field name	[Event type]
size (char)	1
type	char
description	Describes type of event or alarm.
field name	[Event label]
size (char)	16
type	character
description	A string of characters used to describe the event on the screen.
field name	[Event time]
size (char)	6
type	timestamp (hexidecimal)
description	Time of event since TRAIN PRESENT signal was engaged.

NOTE: This message may require further revision to allow more complete care information to be conveyed with the basic axle alarm data. The device creating the alarm may also be one reading the car tag or counting the cars.

Message 'S': Train Summary

The Train Summary data message provides a means for one detector system to send text data through another system for display at a remote location without the intermediate system taking an active part in processing or handling the information. This data message consists of the following fields:

[mtype] { summary text }

field name	[mtype]
size (char)	1

type	constant
description	'S'
field name	{ summary text }
size (char)	undetermined
type	character
description	Train data to be stored and displayed upon command from the remote destination.

Message 'D': **Detail Train Data**

The Detail Train data message provides a means for one detector system to send text data through another system for display at a remote location without the intermediate system taking an active part in processing or handling the information. This data message consists of the following fields:

[mtype] { detail text }

field name	[mtype]
size (char)	1
type	constant
description	'D'
field name	{ detail text }
size (char)	undetermined
type	character
description	Train data to be stored and displayed upon command from the remote destination.

Message 'N': **No Train**

The No Train command consists of one field:
[mtype]

field name	[mtype]
size (char)	1
type	constant
description	'N'

Message 'Q': **Status Request**

The Status Request command consists of one field:
[mtype]

field name	[mtype]
size (char)	1
type	constant
description	'Q'

Message 'L': **Last Message**

The Last Message command consists of one field:
[mtype]

field name	[mtype]
size (char)	1
type	constant
description	'L'