



## OpenLCB Technical Note

### Train Search Protocol

Jan 24, 2021

Preliminary

## 1 Introduction

This Technical Note provides background information and commentary to the OpenLCB Train Search Protocol Standard (in short ‘the Standard’). This document is not intended to be read standalone, but in parallel to the Standard.

- 5 When we use capitalized terms Train, Throttle, and Command Station, we refer to the definitions according to the Section ‘Interactions’ in the Standard.

### 1.1 Background

#### 1.1.1 Relation to Traction Protocol

- 10 The OpenLCB Network is a peer-to-peer network where Nodes connected to the network can communicate with each other using the OpenLCB Network Protocols. These protocols are organized around the different OSI layers, and to achieve an end-to-end use-case a multitude of protocols are typically used, from different layers<sup>1</sup>, or in a complementary manner<sup>2</sup>.

- 15 Some of these protocols are **broadcast**, meaning that all Nodes receive the messages of the interaction; other protocols are **addressed**, meaning the interaction takes place between two specific Nodes, and other Nodes do not receive the messages of the interaction. In order for an addressed interaction to take place, the initiating Node needs to know the Node ID of the target Node. This generally requires some broadcast protocol using which the initiating Node can discover the possible target Nodes, and present these on a user interface to the user to select.

- 20 Considering the interaction of a user using an OpenLCB-connected Throttle to communicate to an OpenLCB-connected Train, or an OpenLCB-connected Command Station, we have two protocols that are part of this interaction:

- The Traction Protocol defines the addressed interactions related to driving a train, such as setting speed and direction, or operating functions. It takes place between one Throttle and one or more Train Node(s) that are controlled.
- 25 • The Train Search Protocol is a broadcast protocol that specifies interactions needed for the Throttle to identify the Node ID of the Train Node that the user intends to drive.

#### 1.1.2 Relation to Legacy Track Protocols

Trains may or may not be equipped with the necessary technology to directly participate in the OpenLCB Network. For now and the foreseeable future, an overwhelming majority of digitally

<sup>1</sup>Example: The Memory Configuration Protocol is using the Datagram Protocol as a supporting layer.

<sup>2</sup>Example: for end users to configure an OpenLCB Node, the Memory Configuration Protocol is expanded with the Configuration Description Information protocol to describe what configuration settings are available at which memory offsets.

30 controlled trains will be using a mobile decoder that operates using an existing standard or proprietary  
method of transmitting commands to the mobile decoder through the track, such as the NMRA®  
DCC® protocol suite.

In such a setting, there is a centralized component called Command Station which is responsible for  
generating the track protocol signal. This device has to receive commands from the Throttles, and  
35 translate them to remote control the mobile decoders built into the trains rolling on the track. The  
Command Station is then responsible for participating on the OpenLCB Network and representing the  
remote trains. The Traction Protocol requires each train to be a separate OpenLCB Node with its own  
Node ID, therefore the one Command Station device will be operating a number of Virtual Nodes, one  
per actively controlled train. All of these Virtual Train Nodes participate in the Train Search Protocol  
40 so that Throttles can discover, find and operate the trains connected to the track.

An important question then becomes: how does the Command Station know when the user intends to  
address a new train on the track that does not yet have a matching Virtual Train Node. The Train  
Search Protocol addresses this use-case as well.

### 1.1.3 Relation to other Search and Discovery Protocols

45 Note that the existence of Train Search Protocol does not exclude other methods and standards,  
including some not yet developed, for the same or similar purposes. There are also possibilities of using  
less capable existing Standards to achieve similar results (covering some but not all use-cases), albeit at  
a higher cost for the Throttle Node in terms of the necessary resources (above all RAM) and higher  
number of network messages to send and keep track of.

50 Some of these options are described in the Alternatives Considered section below (TODO: replace with  
reference).

## 1.2 Served Use Cases

- The User is holding a Throttle connected to the OpenLCB Network. The Throttle has a numeric  
keypad and no screen. The user types in the four digit cab number they see on the locomotive  
55 and presses the Enter key. The Throttle uses the Train Search Protocol to find the matching  
OpenLCB Node. Then the throttle uses the Traction Protocol to connect to that Train Node, and  
the user is able to drive the train.

Note that this use-case presents the same way if the train itself has a radio-equipped decoder  
and is directly participating on the OpenLCB network; and also if the train has a DCC decoder  
60 and represented by a Command Station on the OpenLCB Network.

- The user owns a locomotive with road number 474 014 with a legacy track protocol decoder.  
The user does not remember what address is assigned to the decoder, but there is a Roster  
(locomotive database) in the Command Station, which has a record of the track address, and the  
name “Re 474 014” is assigned to this record. The user has a Throttle with a numeric keypad  
65 and a screen, and types in 474 on the keypad. The Throttle uses the Train Search Protocol to  
look for matching entries on the network. The Command Station searches through the Roster,  
instantiates the matching Virtual Nodes, provides the list of matching Node IDs to the Throttle.  
The Throttle queries the user-visible names of those Nodes using the Simple Node Information  
Protocol and presents a list to the user to select from. The user selects “Re 474 014”, and starts  
70 driving the train.

- The user purchased a new locomotive which came with a Marklin-Motorola format mobile decoder with address 72. The user puts it onto the track, types in 72 on the throttle, selects the Marklin-Motorola track protocol using option keys, and presses Enter. The user is in control of the locomotive.
- 75 • A friend comes over and brings a DCC-equipped locomotive with them. They warn that the locomotive has a very old decoder. The user enters the address on the Throttle, selects the DCC-14-speed-step protocol using option keys, and presses Enter. The user is in control of the locomotive.
- 80 • A Gateway device connects a proprietary layout control bus to OpenLCB. Throttles connected to the proprietary bus select locomotives by a DCC address. The Gateway device intercepts these messages and sends out Train Search Protocol messages to the OpenLCB bus. A DCC Command Station connected to the OpenLCB bus creates Virtual Train Nodes for the intended locomotives. A radio-equipped OpenLCB locomotive that has an up to 4 digit number on the cab can also be selected using this method.
- 85 • A large software system designed to control model railroads has a system-specific implementation module for a variety of different command systems. The API of this module gives a 4-digit DCC address to select a locomotive. The OpenLCB implementation of this API translates this number to a Train Search Protocol message in order to trigger an OpenLCB-connected Command Station to create a Virtual Node. The Command Station uses its settings to
- 90 decide whether it should be a DCC protocol train or some other protocol (e.g. Marklin-Motorola). A radio-equipped OpenLCB locomotive that has an up to 4 digit number on the cab can also be selected using this method.

### 1.3 Unserved Use Cases

- 95 • The Train Search protocol is not a generic search protocol. It is not possible to search for any other entity than a Node<sup>3</sup>. While the concept could be used for searching for other types of nodes, the specific allocation shall only be used for searching for Train Nodes. This is necessary in order to ensure that Throttles that use this allocation properly function on the network.
- 100 • The search results are only Node IDs. If additional information beyond the Node ID is needed (such as user-visible name), the Throttle has to use other protocols (such as the Simple Node Identification Protocol) to query that information.
- Only digits are supported in the search query. The maximum length of the query is 6 digits. Longer and alphanumeric search queries are not supported.
- 105 • A Command Station can only receive a signal using the Train Search Protocol when the user wants a new Virtual Train Node to be created. The Train Search Protocol does not support giving a signal to the Command Station on when to *delete* a Virtual Train Node.

### 1.4 Overview of the Protocol

The Train Search Protocol is fundamentally a broadcast search protocol in a peer-to-peer network. The Throttle broadcasts the search query, which reaches all other Nodes; the matching Nodes reply with a broadcast message which reaches the Throttle, allowing the Throttle to compile a list of results.

<sup>3</sup>For example it is not possible to search for an Event.

110 The input from the Throttle side is a number with up to 6 digits. This number is encoded using BCD into 3 bytes. An additional flag byte is appended that describes options related to protocol to be used (e.g. DCC with 128-speed-step) and specifies certain behavioral parameters for what Train Nodes should report a match. A fixed 4-byte prefix is prepended, which yields an Event ID.

115 The Throttle sends out an Identify Producer (broadcast) message to the bus. This broadcast message will reach all segments where there is a Command Station or a native OpenLCB Train Node<sup>4</sup>.

Train Nodes (both OpenLCB-connected and Command Station managed), upon receipt of such Identify Producer command, will evaluate the request against their identifying properties (e.g. address, cab number, Node name). If they find that the request matches their identifying properties, the Node responds with a Producer Identified message on the same Event ID. If the request does not match that node, no Producer Identified message is emitted.

If no response arrives for 200 msec, and the request specifies so, then a Command Station will create a new Virtual Train Node matching the parameters in the request. The new Node will then emit a Producer Identified message to the bus.

125 The Throttle is waiting for Producer Identified messages on their requested Event ID. These messages contain a Source Node ID (or equivalent representation). These are the search results. The Throttle may pick the first search result's Node ID and operate that train using the Traction Protocol, or if multiple responses arrive, then the Throttle may interrogate those Nodes using the SNIP and PIP protocol to present a list of possible trains to select to the user, if there is a display.

## 2 Annotations to the Standard

130 The Subsections herein mirror the structure of the Train Search Protocol Standard document.

### 2.1 Introduction

Note that this section of the Standard is informative, not normative.

135 The Train Search Protocol is built on top of the Event Transport Protocol, which operates using generic OpenLCB messages. This means that the Train Search Protocol can operate on any supported OpenLCB transport layer, including CAN-bus and TCP. This allows both wired and wireless Throttles to operate using the Train Search Protocol.

### 2.2 Intended Use

Note that this section of the Standard is informative, not normative.

### 2.3 References and Context

140 There is no reference to the Traction Protocol, which the Throttle would use to drive actual trains. This is on purpose, because the scope of the Train Search Protocol ends after the Throttle discovers the Node ID. The scope of the Traction Protocol begins once a Node ID is known.

10 <sup>4</sup>Command Stations and Train Nodes shall declare themselves by sending a Producer Range Identified message upon startup that covers the entire range of the Train Search Protocol Event IDs.

## 2.4 Message Formats

145 This Standard is built on top of the Event Transport Standard. All of the necessary messages to operate this protocol are allocated and defined in that Standard.

## 2.5 Allocation

### 2.5.1 Identifier Range allocation and License terms

150 The Train Search Protocol was developed for and proposed as a Standard by Train Control Systems, Inc. It is a valid use of the Unique Identifiers allocated to any specific party for that party to develop proprietary protocols for operating on the OpenLCB network. It is also valid for this party to offer the developed protocol with appropriate documentation as a proposed Standard.

The licensing terms attached to the Standard allows anyone to create compatible products without any concerns related to license fees. It is not permissible however to create an incompatible product.

### 2.5.2 Identifier Format

155 This section only defines what the individual bits and bytes of the given identifiers are allocated to. It is described in the Section 'Interactions' how the different parties should be using those fields and identifiers.

Notes about the symbols used in the Track Protocol table:

- 0b10 is a binary number of size 2 bits. The MSB is 1, the LSB is 0.
- 160 • The symbol " signals a repeat of the previous row's value in this column.
- The symbol \* signals that the row applies to all possible values in this field.

## 2.6 Interactions

This section contains a glossary of terms used in the description of the interactions. These terms are used throughout the entire Standard and Technical Note according to these definitions.

165 Examples:

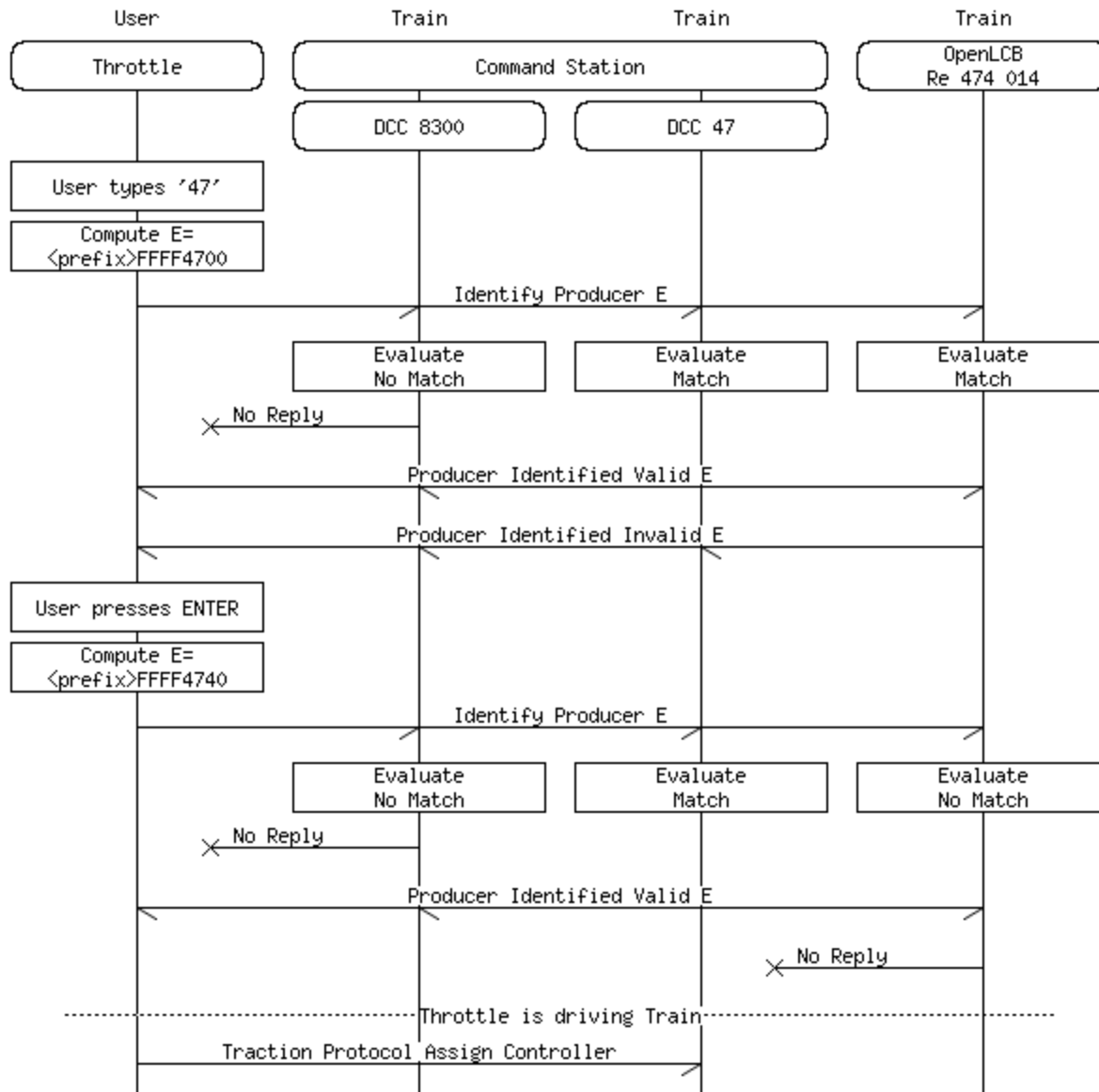
- A Train Node could be a WiFi-connected decoder built into a model locomotive, which establishes a connection to and participates in the OpenLCB Network. Alternatively, it could be a Virtual Train Node represented by a DCC Command Station for the purpose of the OpenLCB bus.
- 170 • The Command Station might be a DCC Command Station, creating the track signal. It could also be a Gateway from OpenLCB to a proprietary bus, remote controlling a Command Station connected to that proprietary bus.

Examples for Train Node properties:

- The Name might be
  - 175 ○ "Re 474 014-5" for a Swiss electric locomotive of the series 474;
  - "BR18 313" for a German steam engine of the series 18;

- “C&NW F7A 415” for an F7 unit of the Chicago&Northwestern Railroad.
- “Thomas the Tank Engine”.
- Protocol: DCC 7-bit address; DCC 14-bit address; Marklin-Motorola; MFX/M4; OpenLCB.
- 180 • Protocol Version: DCC has three different speed step modes (14, 28, 128 speed step mode). Marklin-Motorola has multiple versions.
- 185 • Address: 1 to 127 for DCC 7-bit, 0 to 10239 for DCC 14-bit, 1 to 80 (or 255) for Marklin-Motorola. There is no address for OpenLCB and for MFX/M4, because for these protocols the user is not required to configure a unique numeric value for each decoder. The manufacturer-assigned unique addresses are not relevant for the user and we should not expect the user to be aware or ever input those addresses by the number. It might be beneficial for an OpenLCB-connected mobile decoder to allow the user to configure a purely numeric cab number, which the Train Node may apply as if it was a DCC address. This can enable a DCC-only throttle to select that OpenLCB locomotive instead of a DCC locomotive of that address. The user has to be aware that such an option makes that specific DCC address unavailable, as the Command Station will never instantiate a Virtual Train Node for that address.
- 190

### 2.6.1 Search for existing train nodes

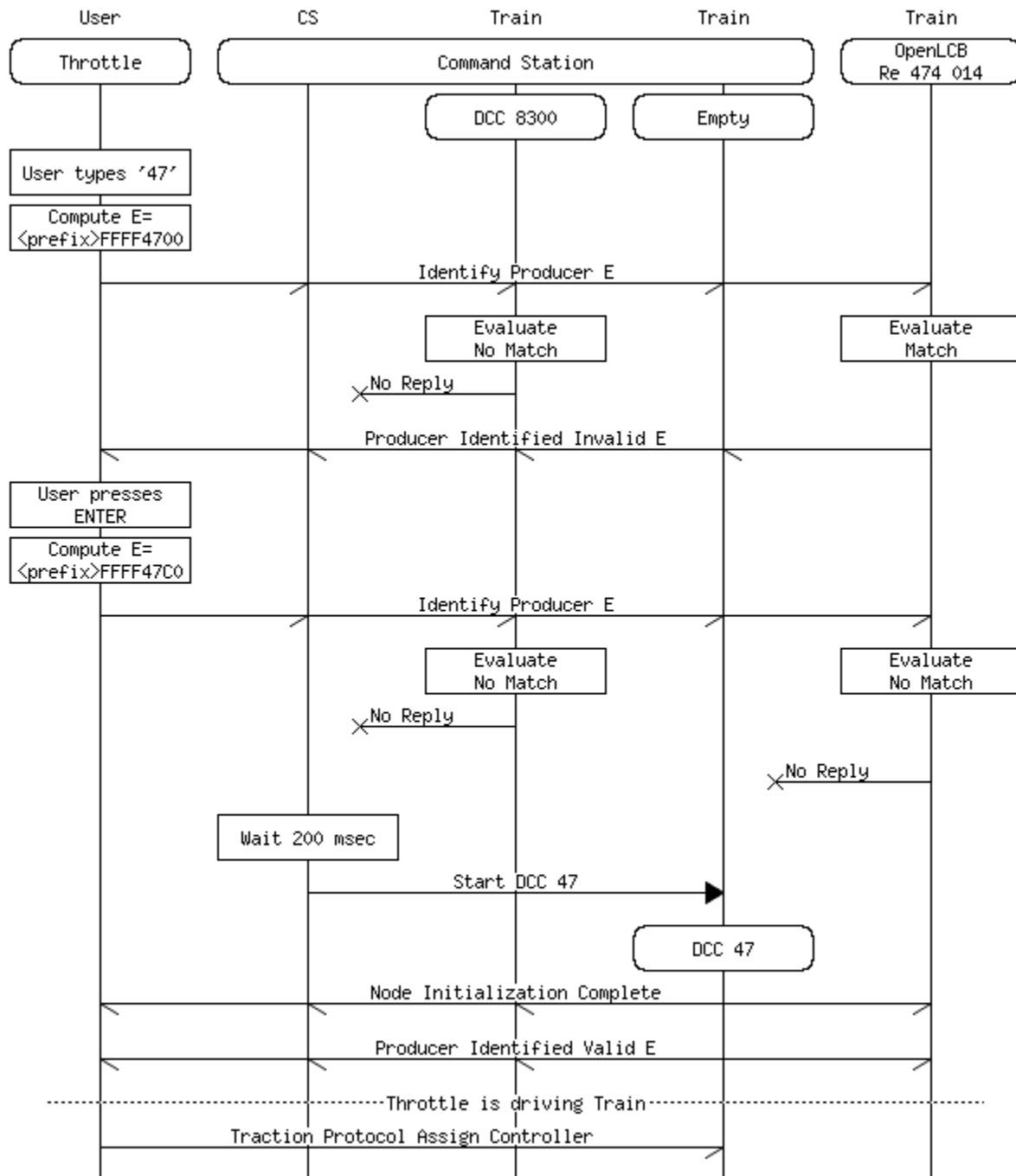


On this diagram we see two interactions with the Train Search Protocol.

In the first interaction the user enters some digits, and the Throttle sends this out as a prefix search for trains starting with “47”. There are multiple replies coming back.

In the second interaction the user presses ENTER, and the Throttle sets the “Exact” bit in the flags byte of the request. Only one reply is generated.

## 2.6.2 Allocate a new Train Node



- 200 On this diagram we see the same two interactions, where the DCC 47 node does not exist at the beginning of the interaction. The Throttle is configured to send the request with the “Allocate” bit set when the user pressed ENTER. The Command Station waits for 200 msec for any responses to come in to the exact match request, after which it assigns an empty Train Node to proxy for DCC Address 47. The newly started node then responds to the Throttle.



205 If the low five bits of the flag byte contain a protocol selection information, then the Command Station would create the new Virtual Train Node according to that protocol. Here the CS used the default settings.

### 2.6.3 Event Identifier matching algorithm

### 2.6.4 Search Result Differentiation

## 210 3 Alternatives considered

- A more generic protocol could be developed for searching for Nodes on the network; this may be peer-to-peer or based on a centralized database built and maintained by a single Node that stores details about all Nodes available on the network (including all Train Nodes).

215 At the time of this writing, no specific proposals exist for this protocol. The requirements space and ambiguity in this domain is very large, and initial discussions in the OpenLCB Group have not sufficed to identify a straightforward solution to this problem.

- All Nodes can be enumerated by the unaddressed Verify Node ID Global message. There is no throttling mechanism provisioned to maintain the list of replies with a limited amount of memory. The Node list can then be further processed using Protocol Support Inquiry (PIP) messages to restrict to Train Nodes and Simple Node Information Request (SNIP) messages for querying identifying properties such as train name.

225 Without a throttling mechanism, using this method to find and identify Train Nodes would require an unbounded amount of memory for the Throttle Nodes. This makes it difficult to build simple and cost efficient Throttles.

- All Train Nodes can be identified using the Well-Known Event ID IsTrain. There is no throttling mechanism provisioned to maintain the list of replies with a limited amount of memory.

230 Without a throttling mechanism, using this method to find and identify Train Nodes would require an unbounded amount of memory for the Throttle Nodes. This makes it difficult to build simple and cost efficient Throttles.

- A legacy track protocol Command Station could expose a command interface either via dedicated OpenLCB messages or via a Datagram-based protocol to maintain its list of virtual Train Nodes. This protocol could also be out-of-band, for example through a user interface (physical or HTTP-based), or could be coupled to a persistent database using the Memory Configuration Protocol and the Configuration Descriptor (CDI) XML to expose the user interface via a Configuration Tool.

240 There was an actual protocol proposal developed for this alternative under the name of Traction Proxy Protocol. The difficulty with this approach is that a Throttle can be programmed to operate using the Traction Proxy Protocol, which would make the Throttle work well with a Command Station, but naturally exclude the ability of this Throttle to connect to native OpenLCB Train Nodes. This makes the solution not desirable as a long-term offering, as it does not provide a seamless upgrade path to radio-equipped locomotives.

- Instead of encoding the information to transmit in Event IDs, a protocol could be developed using custom OpenLCB messages that are marked as global/broadcast in the MTI. This would allow fewer compromises around the encoding.

Creating a new broadcast protocol would pose difficult questions around buffering and traffic routing. The Event Transport Standard has specific preparation about interconnecting different network segments with filtering and routing for example. All of these features would need to be specifically duplicated to reach a similar performance characteristics as using Events. The duplicate features would then have to be specifically implemented by all gateways and routers.

Even with a protocol defining custom messages, there would be a fairly strict limit on the amount of payload bytes that can be transported. This is because there is no generic protocol layer for transporting broadcast messages on CAN-bus with more than 8 bytes of payload.

## 4 Test Vectors

## 5 Section Title

Quisque sollicitudin tempor bibendum. Donec consectetur condimentum sollicitudin. Sed dignissim velit id felis lacinia at eleifend nisi laoreet. Vivamus tristique porta ornare. Vivamus feugiat dolor id lectus aliquet luctus.

Quisque sollicitudin tempor bibendum. Donec consectetur condimentum sollicitudin. Sed dignissim velit id felis lacinia at eleifend nisi laoreet. Vivamus tristique porta ornare. Vivamus feugiat dolor id lectus aliquet luctus.

Quisque sollicitudin tempor bibendum. Donec consectetur condimentum sollicitudin. Sed dignissim velit id felis lacinia at eleifend nisi laoreet. Vivamus tristique porta ornare. Vivamus feugiat dolor id lectus aliquet luctus.

Quisque sollicitudin tempor bibendum. Donec consectetur condimentum sollicitudin. Sed dignissim velit id felis lacinia at eleifend nisi laoreet. Vivamus tristique porta ornare. Vivamus feugiat dolor id lectus aliquet luctus.

Quisque sollicitudin tempor bibendum. Donec consectetur condimentum sollicitudin. Sed dignissim velit id felis lacinia at eleifend nisi laoreet. Vivamus tristique porta ornare. Vivamus feugiat dolor id lectus aliquet luctus.

## Table of Contents

1	Introduction.....	1
1.1	Background.....	1
1.1.1	Relation to Traction Protocol.....	1
1.1.2	Relation to Legacy Track Protocols.....	1
1.1.3	Relation to other Search and Discovery Protocols.....	2
1.2	Served Use Cases.....	2
1.3	Unserved Use Cases.....	3
2	Annotations to the Standard.....	3
2.1	Introduction.....	4
2.2	Intended Use.....	4
2.3	References and Context.....	4
2.4	Message Formats.....	4
2.5	Allocation.....	4
2.5.1	Identifier Range allocation and License terms.....	4
2.5.2	Identifier Format.....	4
2.6	Interactions.....	4
2.6.1	Search for existing train nodes.....	4
2.6.2	Allocate a new Train Node.....	4
2.6.3	Event Identifier matching algorithm.....	4
2.6.4	Search Result Differentiation.....	4
3	Alternatives considered.....	4
4	Test Vectors.....	6
5	Section Title.....	6