TELEMETRY GROUP

129-22

TELEMETRY NETWORK STANDARD HANDBOOK

DOCUMENT 129-22

TELEMETRY NETWORK STANDARD HANDBOOK

February 2022

Prepared by

TELEMETRY GROUP

Published by

Secretariat
Range Commanders Council
US Army White Sands Missile Range,
New Mexico 88002-5110

Table of Contents

Table of Figures

Table of Tables

This page intentionally left blank.

Preface

This document is a product of the Range Commanders Council Telemetry Group's task TG-175. The objective of the task was to develop a handbook/user's guide leveraging the expertise of the iNET lead developer. The handbook is to feature how to/tips/cautions and lessons learned during iNET development and testing. The current TmNS Chapter 23 in IRIG 106 contains some MDL examples; those are to be ported over to the handbook, along with some other sections of the TmNS that are published there now because there was no other place to maintain the information.

Since the MDL is also a new entity, the plan is to include as many MDL examples as time and budget constraints will allow. It is expected that this first edition of a Telemetry Networks Handbook will not be fully complete and will expand as the range user base gains experience and discovers information warranted for the handbook.

Contact the following for questions about this document.

Secretariat, Range Commanders Council
ATTN: TEWS-TDR
1510 Headquarters Avenue
White Sands Missile Range, New Mexico 88002-5110
Telephone:(575) 678-1107, DSN 258-1107
E-mail: rcc-feedback@trmc.osd.mil

This page intentionally left blank.

Acronyms

A2A	antenna-to-antenna
ACU	antenna control unit
COTS	commercial off-the-shelf
CRC	cyclic redundancy check
DAU	data acquisition unit
DSCP	DiffServ Code Point
GMC	grandmaster clock
GPS	Global Positioning System
GUI	graphical user interface
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IP	Internet Protocol
LDPC	low-density parity check
LM	link manager
LTC	Latency/Throughput Critical
MAC	media access control
MCR	mission control room
MDID	Message Definition ID
MDL	Metadata Description Language
MIB	management information base
ms	millisecond
N2N	network-to-network
PCM	pulse code modulation
PDID	Package Definition ID
PTP	Precision Time Protocol
QoS	Quality of Service
R2R	range-to-range
RAN	radio access network
RC	Reliability Critical
RF	radio frequency
RFC	Request for Comments
RFNM	RF network message
RSSI	received signal strength indication
RT	remote terminal
SIP	Session Initiation Protocol
SNMP	Simple Network Management Protocol
SST	serial streaming telemetry
SysMgr	system manager
TA	test article
TCP	Transmission Control Protocol
TDM	TmNS Data Message
TDMA	time division multiple access
TMA	TmNS manageable application

TmNS	Telemetry Network Standard
TSS	TmNS Source Selector
UDP	User Datagram Protocol
VoIP	Voice over Internet Protocol

CHAPTER 1

Introduction to the TmNS Handbook

The Telemetry Network Standard (TmNS) defined in Chapters 21-28 of IRIG 106 provides the framework for which network-based telemetry systems are built. The TmNS leverages existing standards and protocols as well as defines some of its own in order to meet the unique requirements of the flight test domain. These standards identify interfaces for configuration, management, network transport protocols, a two-way telemetry link, and various other system and component capabilities. On its own, the TmNS does not define a system; rather, it is a toolbox that contains everything needed to build a TmNS-based system. The purpose of this handbook is to provide guidance to a system designer for how to use the available tools of the TmNS in order to build a TmNS-based system that meets the capabilities of the system. This includes the process of component selection (generic, not vendor-specific) and configuration in order to achieve the desired system capability.

This handbook is written with two primary audiences in mind: system designers and component developers. The system designers are primarily interested in the interfaces, protocols, and components necessary to field a particular system capability. On the other hand, component developers will be most interested in seeing Metadata Description Language (MDL) example configurations of capabilities that their components will need to implement. This handbook is organized by various system-level capabilities, allowing system designers to easily understand what components and capabilities are required in order to realize the system-level capability of interest. For each capability, MDL examples are provided as a reference for component developers for the specific component capabilities that accompany the system-level capability.

Chapter 2 describes TmNS components and their capabilities and functions. Some of these capabilities are common to TmNS components, and some are component-specific. General definitions of components and their collection of capabilities are given for the sake of discussion throughout this document, such as a list of specific capabilities that make up a data acquisition unit (DAU). These components are identified up front and are referenced by system capabilities in following sections that utilize these components.

Chapter 3 through Chapter 8 represent different system-level capabilities that can be achieved with a TmNS-based system. These capabilities can be combined, but they do not have to be. Some capabilities, though, will depend upon other capabilities in order to achieve them (e.g., a Voice over Internet Protocol [VoIP] capability also requires a two-way telemetry capability). As such, it is the intent of this document to treat these system capabilities as separate, stand-alone capabilities when possible. The intent of these chapters is to identify the specific capabilities and everything needed in order to achieve that capability.

The system-level capabilities identified included the following.

- Data Acquisition Capability
- Two-Way Telemetry Capability
- Instrumentation Control Capability
- Data Retrieval Capability
- Voice Over Internet Protocol (IP) Capability

- Test Article (TA)-to-TA Relay Capability

Some of the identified capabilities may have multiple possible implementations. This handbook aims to guide the system designer to the appropriate subsection that meets the needs of the system being designed.

An example MDL configuration file will be linked throughout the document in order to serve as a reference implementation for the capability or component being discussed. While the MDL schema may be flexible enough to allow different implementations of the same feature, the examples are provided to serve as the best practices approach for the capability. Following these best practices will increase the likelihood of component interoperability and successful integration into the TmNS-based system. Each capability section contains one or more links to example MDL files that describe the relevant capability. A screenshot of the relevant portions of MDL for the capability is also included for quick reference. Vendors and component developers will likely benefit the most from these examples. While MDL configuration is a key aspect of TmNS-based systems, the handbook makes no attempt to turn a system designer into an MDL expert. The linked files are provided primarily for the component developers.

Chapter 9 provides some simple guidance for basic system troubleshooting and system verification. It also contains a subsection for general lessons learned and other tips for designing and/or troubleshooting systems.

Appendix A identifies additional networking concepts. This appendix is a high-level collection of definitions, concepts, and references for handbook users to turn to for more background knowledge on the underlying technologies that comprise the TmNS. System designers are expected to have a working knowledge of network-based systems. The appendix is meant to serve as a resource for general networking concepts and a high-level view of TmNS concepts.

The example test configurations and MDL files presented throughout this handbook are meant to provide a basic level of guidance to test designers and component implementers. Figure 1-1 provides a range-level view of a TmNS-based system (or collection of TmNS-based systems) from which the various MDL file examples in the handbook correspond. Specific system capabilities will only involve subsets of the range-level system, and each example will highlight the relevant portions of the system required for providing the capability. The figure is meant to provide some context for the examples throughout this handbook.

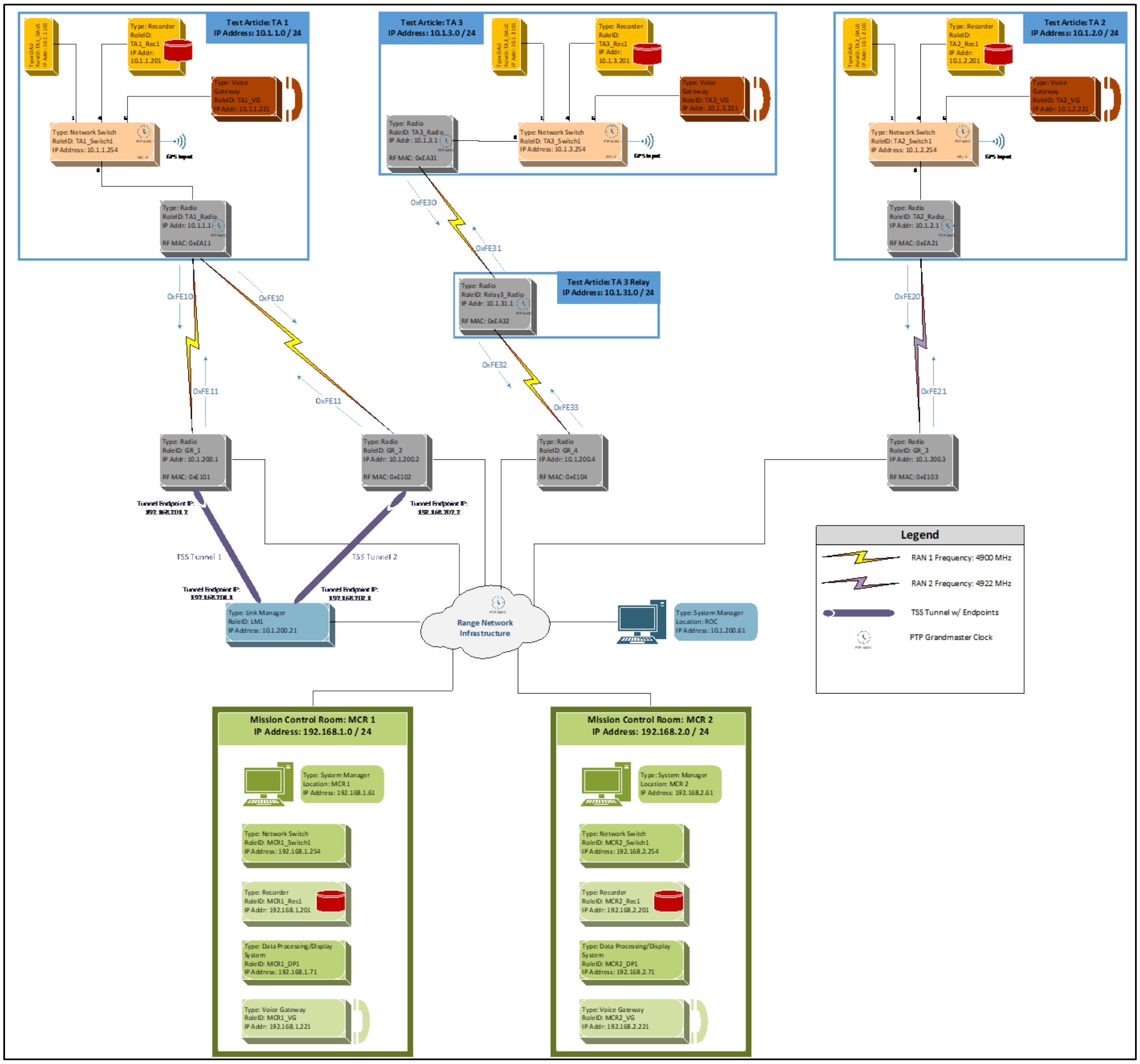

Test Article: TA 1
IP Address: 10.1.1.0 / 24
Type DAU RoleID: TA1_DAU1 IP Addr: 10.1.1.101
Type: Recorder RoleID: TA1_Rec1 IP Addr: 10.1.1.201
Type: Voice Gateway RoleID: TA1_VG IP Addr: 10.1.1.221
Type: Network Switch RoleID: TA1_Switch1 IP Address: 10.1.1.254
GPS Input
Type: Radio RoleID: TA1_Radio IP Addr: 10.1.1.1 RF MAC: 0xEA11
Test Article: TA 3
IP Address: 10.1.3.0 / 24
Type DAU RoleID: TA3_DAU1 IP Addr: 10.1.3.101
Type: Recorder RoleID: TA3_Rec1 IP Addr: 10.1.3.201
Type: Voice Gateway RoleID: TA3_VG IP Addr: 10.1.3.221
Type: Radio RoleID: TA3_Radio IP Addr: 10.1.3.1 RF MAC: 0xEA31
Type: Network Switch RoleID: TA3_Switch1 IP Address: 10.1.3.254
GPS Input
Test Article: TA 2
IP Address: 10.1.2.0 / 24
Type DAU RoleID: TA2_DAU1 IP Addr: 10.1.2.101
Type: Recorder RoleID: TA2_Rec1 IP Addr: 10.1.2.201
Type: Voice Gateway RoleID: TA2_VG IP Addr: 10.1.2.221
Type: Network Switch RoleID: TA2_Switch1 IP Address: 10.1.2.254
GPS Input
Type: Radio RoleID: TA2_Radio IP Addr: 10.1.2.1 RF MAC: 0xEA21
0xFE30
0xFE31
Test Article: TA 3 Relay
IP Address: 10.1.31.0 / 24
Type: Radio RoleID: Relay3_Radio IP Addr: 10.1.31.1 RF MAC: 0xEA32
0xFE32
0xFE33
0xFE10
0xFE10
0xFE11
0xFE11
0xFE20
0xFE21
Type: Radio RoleID: GR_1 IP Addr: 10.1.200.1 RF MAC: 0xE101
Type: Radio RoleID: GR_2 IP Addr: 10.1.200.2 RF MAC: 0xE102
Type: Radio RoleID: GR_4 IP Addr: 10.1.200.4 RF MAC: 0xE104
Type: Radio RoleID: GR_3 IP Addr: 10.1.200.3 RF MAC: 0xE103
Tunnel Endpoint IP: 192.168.201.2
Tunnel Endpoint IP: 192.168.202.2
TSS Tunnel 1
TSS Tunnel 2
Tunnel Endpoint IP: 192.168.201.1
Tunnel Endpoint IP: 192.168.202.1
Type: Link Manager RoleID: LM1 IP Address: 10.1.200.21
Range Network Infrastructure
Type: System Manager Location: ROC IP Address: 10.1.200.61
Legend
RAN 1 Frequency: 4900 MHz
RAN 2 Frequency: 4922 MHz
TSS Tunnel w/ Endpoints
PTP Grandmaster Clock
Mission Control Room: MCR 1
IP Address: 192.168.1.0 / 24
Type: System Manager Location: MCR 1 IP Address: 192.168.1.61
Type: Network Switch RoleID: MCR1_Switch1 IP Address: 192.168.1.254
Type: Recorder RoleID: MCR1_Rec1 IP Addr: 192.168.1.201
Type: Data Processing/Display System RoleID: MCR1_DP1 IP Address: 192.168.1.71
Type: Voice Gateway RoleID: MCR1_VG IP Addr: 192.168.1.221
Mission Control Room: MCR 2
IP Address: 192.168.2.0 / 24
Type: System Manager Location: MCR 2 IP Address: 192.168.2.61
Type: Network Switch RoleID: MCR2_Switch1 IP Address: 192.168.2.254
Type: Recorder RoleID: MCR2_Rec1 IP Addr: 192.168.2.201
Type: Data Processing/Display System RoleID: MCR2_DP1 IP Address: 192.168.2.71
Type: Voice Gateway RoleID: MCR2_VG IP Addr: 192.168.2.221

Figure 1-1. Consolidated Range-Level View of Examples Found in the Handbook

This page intentionally left blank.

CHAPTER 2

TmNS Component Capabilities

This chapter identifies the interfaces and capabilities available on TmNS components. There are several different types of TmNS components, each with their own capabilities. The TmNS focuses on defining the capabilities rather than defining specific components. Component specification documents are generally developed during a procurement cycle to identify specific capability requirements to be provided by a particular hardware component. For the sake of discussion throughout this document, some of the more common groupings of capabilities are assigned to a particular component (e.g., a DAU, a recorder, a radio, etc.). Though it is possible that a component can be built that provides a mixture of capabilities that do not exactly meet the component definitions set forth in this document, the general idea behind the components identified in this document are such that the associated component capabilities are required for those system capabilities that use them. Thus, if a single component contains both DAU capabilities and recorder capabilities within a single hardware platform, it may satisfy the requirements of both DAUs and recorders.

A TmNS-compliant component implements a suite of interfaces and protocols according to the capabilities that the component provides. Some of the capabilities are common across all TmNS components, and some are capability-specific. The common and specific capabilities are discussed in the sections below and are followed by the typical capability groupings of TmNS components that are referenced throughout the handbook.

2.1 Common Capabilities of TmNS Components

Most TmNS components maintain a common set of interfaces and capabilities, particularly with regard to system management and configuration. This section identifies three key common TmNS interfaces and how they are utilized. This includes network interfaces, system management interfaces, and the MDL configuration interface.

2.1.1 Network Interface

It should come as no surprise that components in a network-based system would have at least one network interface. Network interfaces include wired and/or wireless interfaces. The network interfaces provide the standardized approach for device communication. This includes system management, configuration file delivery, and data transport, among others. Specific networking technologies are highlighted in Chapter 22.[1]

2.1.2 System Management Interface

Generally, TmNS components include a TmNS manageable application (TMA) The TMA provides a standardized interface for monitoring and managing the component. Command and control communications are also carried out through this interface via the network. While Chapter 25[2] identifies two primary protocols for system management, only one is required to be implemented by the component. Simple Network Management Protocol (SNMP) and Hypertext

[1] Range Commanders Council. "Network-Based Protocol Suite." In *Telemetry Standards*. RCC 106-20 Chapter 22. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.
[2] Range Commanders Council. "Management Resources." In *Telemetry Standards*. RCC 106-20 Chapter 25. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

Transfer Protocol (HTTP) are the two named protocols. Care should be taken when selecting a component for use in a system that it implements the system management protocol that is used by the management component in the system, or the device will not be manageable.

NOTE	Early implementations and system demonstrations utilized SNMP version 3 as the management interface.

Chapter 25 also identifies several specific management resources to be implemented by each TMA. For SNMP, these resources have been constructed into the TmNS-management information base (MIB). Some of the resources are common across all TMAs while other resources are only implemented by TMAs whose host platform performs a specific capability or function. For example, all TMAs will implement the same common identifier resources and the same common resources that support the component configuration process, but not all TMAs will implement the resources that will start or stop a recorder.

Aside from those resources identified in Chapter 25 and captured in the TMNS-MIB, many TMAs will also implement other public standard Internet Engineering Task Force (IETF) Request for Comments (RFC) MIBs. The list of these additional public RFC MIBs is found in Chapter 22. Not wanting to reinvent the wheel, the TMNS-MIB only contains those concepts that are unique to TmNS components.

2.1.3 MDL Configuration Interface

All TMAs are configurable through an MDL configuration file. The process to retrieve the configuration file and begin reconfiguration is initiated through the system management interface. Chapter 23[3] defines MDL, and it represents the standardized interface by which component configuration descriptions are exchanged across the system. Many elements within the MDL schema are optional elements, meaning their presence is not required in order for an MDL file to be valid. These optional elements tend to be either generic informational fields or elements related to specific component capabilities that do not apply to all components.

The common MDL configuration interface is a multi-vendor solution. Utilizing MDL as the common mechanism for configuring TmNS components within the system, proprietary vendor software is not required.

2.2 Specific Capabilities of TmNS Components

The TmNS specifies capabilities of components rather than specifying the components themselves. Because of this, it is not always clear what device capabilities exist when someone refers to a particular piece of hardware, such as a recorder. The TmNS identifies several specific functions and capabilities that a compliant device may perform, but it does not provide guidance

[3] Range Commanders Council. "Metadata Configuration." In *Telemetry Standards*. RCC 106-20 Chapter 23. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

as to how those capabilities might be grouped into a hardware platform. Some common groupings of capabilities are described in Section 2.3. Table 2-1 lists the different capabilities as identified in Chapter 25. Each capability has specific TmNS management resources associated with it.

Table 2-1. TmNS-Specific System Management Capabilities

System Management Capability	Description
tmnsNetworkFabricDevice	Status of Internet Group Management Protocol (IGMP), multicast routes, and IEEE 1588 port clock states associated with network transport devices that provide routing and/or switching capabilities.
tmnsACU	Status and controls of capabilities common to antenna control units (ACUs).
tmnsDAU	Data source peripheral capabilities, such as is common to DAUs.
tmnsRecorder	Data sink peripheral capabilities, such as is common to recorders. Includes recording controls and media descriptions and status.
tmnsMasterClock	Status and controls of capabilities common to external time source master clocks.
tmnsSSTTx	Status and controls of capabilities common to serial streaming telemetry (SST) transmitters.
tmnsSSTRx	Status and controls of capabilities common to SST receivers.
tmnsAdapter	Status and controls of capabilities for a TMA that allows for multiple TMAs to be present on a single component.
tmnsRCDataSource	Status of Reliability Critical (RC) data connections including transmission statistics by an RC data source.
tmnsLTCDataSource	Control for enabling the sending of TmNS data messages (TDMs), transmission statistics for bytes and messages sent by a Latency/Throughput Critical (LTC) data source.
tmnsLTCDataSink	Statistics for bytes and messages received and messages not received by an LTC data sink.
tmnsConsolidatedManager	Status and controls of capabilities of a TMA that provides consolidated management for a group of TMAs.
tmnsRadio	Status and controls of capabilities common to radios.
tmnsLinkManager	Status and controls of capabilities common to link managers (LMs).
tmnsRCDataSink	Status of RC data connections including receive statistics by an RC data sink.
tmnsVoiceGateway	Status of capabilities common to a voice gateway.
tmnsTPA	Status and controls of capabilities common to telemetry processor adapters.
tmnsPCMGateway	Status of capabilities common to a pulse code modulation (PCM) gateway, which receives PCM data as an input and can generate associated TDMs onto the network.

tmnsNetworkGateway	Status and controls of capabilities common to network gateways that subscribe to TDMs on the network and produce an associated PCM output.
tmnsRAN	Status and controls of capabilities associated with Quality of Service (QoS) policies used by the radio access network (RAN) components.
tmnsTmnsSourceSelector	Status and controls of capabilities associated with a TmNS Source Selector (TSS) client.

A TMA that supports a particular capability supports all management resources associated with that capability unless explicitly noted.

2.3 Typical Capability Groupings into TmNS Components

As mentioned previously, the TmNS does not define how specific capabilities and functions are grouped into distinct components. Component capabilities are based on specification documents that list the required features of the particular hardware platform. For discussion purposes in this handbook, a handful of components are identified with an associated set of TmNS capabilities mapped to them. While these components are based on initial component development specifications, a component can have any mixture of the specific capabilities defined in the TmNS.

The following sections provide a high-level overview of the hardware capabilities in order to inform the reader what general function(s) the component performs or what service it provides. The details of how these identified components are used are described in the specific system capabilities sections of this handbook.

2.3.1 DAU

A TmNS DAU places acquired data into TDMs and sends them over the network for consumption by recorders, data processing applications, and other devices that subscribe to its stream of TDMs. Data can be acquired through several different types of sensors, including avionics busses. The DAUs may come in different shapes and acquisition capability, but the underlying functions are the same. In this handbook, a TmNS DAU has the system management capabilities and related MDL elements as found in Table 2-2.

Table 2-2. TmNS DAU Capabilities

System Management Resources	MDL Element Tree
tmnsDAU	<TmNSDAU>
tmnsLTCDataSource	<TmNSLTCDataSource>

The previously mentioned example MDL file, here, contains a TmNS DAU.

2.3.2 Recorder

A TmNS recorder is a component that is capable of subscribing to and storing TDMs onto its internal storage media. In general, it is able to receive TDMs as an LTC data sink. A recorder is generally viewed as having some data retrieval capabilities. If so, then the recorder will serve as either an LTC data source or an RC data source for producing data onto the network. Sourcing data on the network is done in response to data requests it has received. In this handbook, a TmNS recorder has the system management capabilities and related MDL elements as found in Table 2-3.

Table 2-3. TmNS Recorder Capabilities

System Management Resources	MDL Element Tree
tmnsRecorder	<TmNSDAU>
tmnsLTCDataSink	<TmNSLTCDataSink>
tmnsLTCDataSource	<TmNSLTCDataSource>
tmnsRCDataSource	<TmNSRCDataSource>

The previously mentioned example MDL file, here, contains a TmNS recorder.

2.3.3 Switch

A TmNS switch provides the network transport for the connected devices, TmNS or not. They provide IEEE 1588-2008 (Precision Time Protocol [PTP] version 2)[4] support to connected devices, ensuring precise time synchronization across the network. These switches may also have the ability to operate as the IEEE 1588-2008 grandmaster clock (GMC) for its network segment. While the grandmaster capability can be obtained by a separate component in the system, this functionality has been incorporated into some switches that contain Global Positioning System (GPS) receivers and can synchronize their internal clocks to GPS. In this handbook, a TmNS switch has the system management capabilities and related MDL elements as found in Table 2-4.

Table 2-4. TmNS Switch Capabilities

System Management Resources	MDL Element Tree
tmnsNetworkFabricDevice	<TmNSNetworkFabricDevice>
tmnsMasterClock	<TmNSMasterClock>

The previously mentioned example MDL file, here, contains a TmNS switch.

2.3.4 Radio

A TmNS radio is a radio frequency (RF) transceiver that enables two-way telemetry. It is an Internet Protocol (IP) version 4 (IPv4) router that joins the RF network with a wired network, either on the TA or on the ground network. The radio also provides a TSS server interface and will operate as a TSS tunnel endpoint when connected to a TSS client, though there are not any manageable resources associated with TSS servers.

In addition to the transceiver-related capabilities mentioned, some radios may have additional capabilities on the platform. A radio may provide a proxy capability for legacy SST transmitters, in which case they present themselves as having the tmnsSSTTx capability and relay the commands and status queries and their responses to and from the legacy transmitter over some other out-of-band communication mechanism, such as a serial port connection. Another TmNS capability that may be supported by some radio platforms includes the tmnsMasterClock capability. This capability may be present if the radio platform contains a GPS receiver and can synchronize its internal clock to GPS.

[4] Institute of Electrical and Electronics Engineers. *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*. IEEE 1588-2008. 24 July 2008. Superseded by IEEE 1588-2019. Retrieved 20 January 2022. Available at https://standards.ieee.org/standard/1588-2008.html.

In this handbook, a TmNS radio has the system management capabilities and related MDL elements as found in Table 2-5.

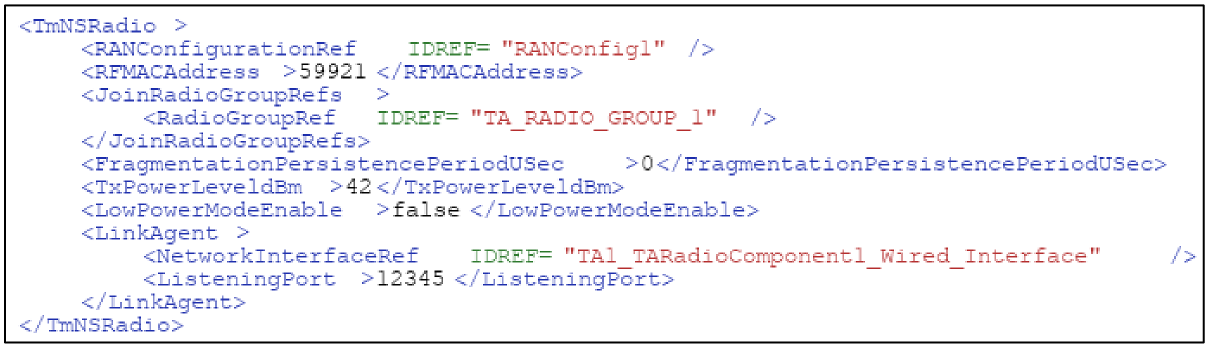
Table 2-5. TmNS Radio Capabilities

System Management Resources	MDL Element Tree
tmnsRadio	<TmNSRadio>
tmnsRAN	<RANConfiguration>
tmnsNetworkFabricDevice	<TmNSNetworkFabricDevice>
tmnsMasterClock	<TmNSMasterClock>
tmnsSSTTx	<TmNSSSTTx>

The previously mentioned example MDL file, here, contains a TmNS radio.

The relevant portions of the MDL file associated with this TmNS radio and its capabilities are included in the following pictures.

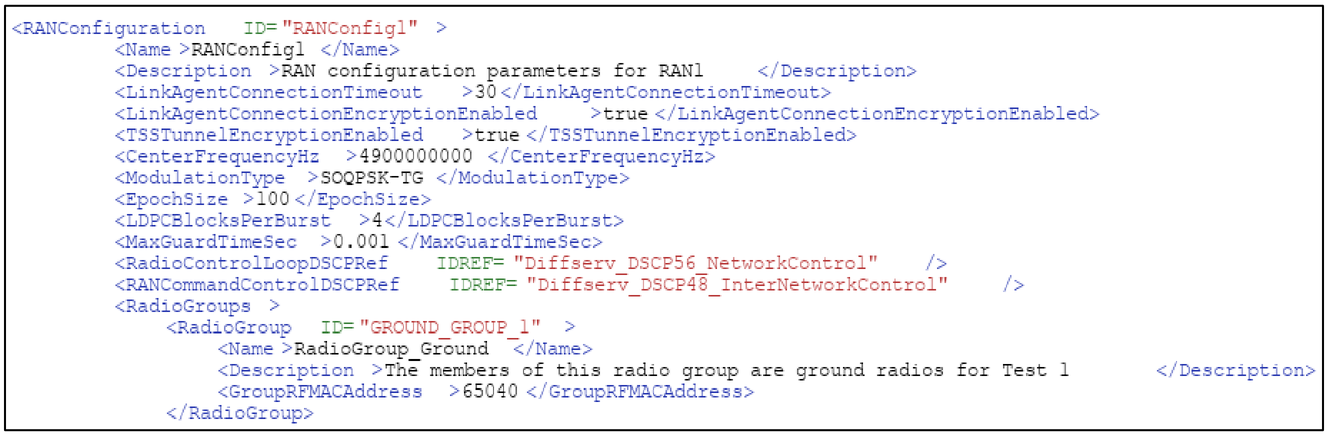
Figure 2-1 shows the MDL configuration elements associated with the TmNSRadio capability.



```
<TmNSRadio >
     <RANConfigurationRef    IDREF= "RANConfig1"  />
     <RFMACAddress  >59921 </RFMACAddress>
     <JoinRadioGroupRefs    >
          <RadioGroupRef   IDREF= "TA_RADIO_GROUP_1"   />
     </JoinRadioGroupRefs>
     <FragmentationPersistencePeriodUSec     >0</FragmentationPersistencePeriodUSec>
     <TxPowerLeveldBm  >42</TxPowerLeveldBm>
     <LowPowerModeEnable   >false </LowPowerModeEnable>
     <LinkAgent  >
          <NetworkInterfaceRef    IDREF= "TA1_TARadioComponent1_Wired_Interface"     />
          <ListeningPort  >12345 </ListeningPort>
     </LinkAgent>
</TmNSRadio>
```

Figure 2-1. TmNSRadio Capability Parameters

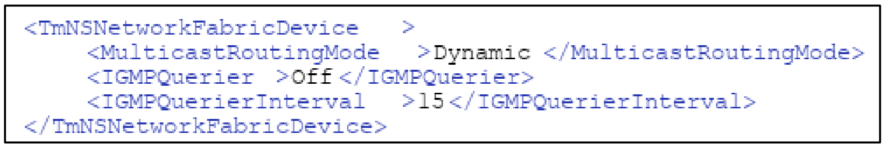
Figure 2-2 shows the MDL configuration elements associated with the RANConfiguration capability.



```
<RANConfiguration   ID="RANConfig1"  >
        <Name >RANConfig1 </Name>
        <Description  >RAN configuration parameters for RAN1     </Description>
        <LinkAgentConnectionTimeout    >30</LinkAgentConnectionTimeout>
        <LinkAgentConnectionEncryptionEnabled     >true </LinkAgentConnectionEncryptionEnabled>
        <TSSTunnelEncryptionEnabled    >true </TSSTunnelEncryptionEnabled>
        <CenterFrequencyHz  >4900000000 </CenterFrequencyHz>
        <ModulationType  >SOQPSK-TG </ModulationType>
        <EpochSize >100</EpochSize>
        <LDPCBlocksPerBurst  >4</LDPCBlocksPerBurst>
        <MaxGuardTimeSec  >0.001 </MaxGuardTimeSec>
        <RadioControlLoopDSCPRef    IDREF= "Diffserv_DSCP56_NetworkControl"    />
        <RANCommandControlDSCPRef    IDREF= "Diffserv_DSCP48_InterNetworkControl"    />
        <RadioGroups  >
            <RadioGroup   ID="GROUND_GROUP_1"   >
                <Name >RadioGroup_Ground  </Name>
                <Description  >The members of this radio group are ground radios for Test 1       </Description>
                <GroupRFMACAddress   >65040 </GroupRFMACAddress>
            </RadioGroup>
```

Figure 2-2. RANConfiguration Capability Parameters

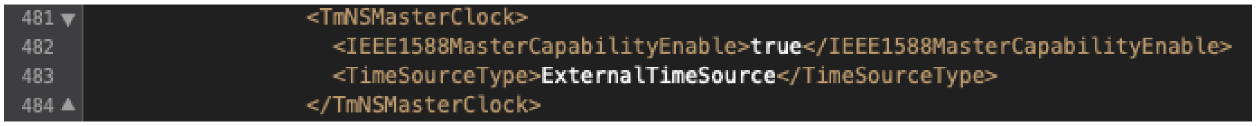
Figure 2-3 shows the MDL configuration elements associated with the TmNSNetworkFabricDevice capability.



```
<TmNSNetworkFabricDevice   >
    <MulticastRoutingMode   >Dynamic </MulticastRoutingMode>
    <IGMPQuerier  >Off </IGMPQuerier>
    <IGMPQuerierInterval   >15 </IGMPQuerierInterval>
</TmNSNetworkFabricDevice>
```

Figure 2-3. TmNSNetworkFabricDevice Capability Parameters

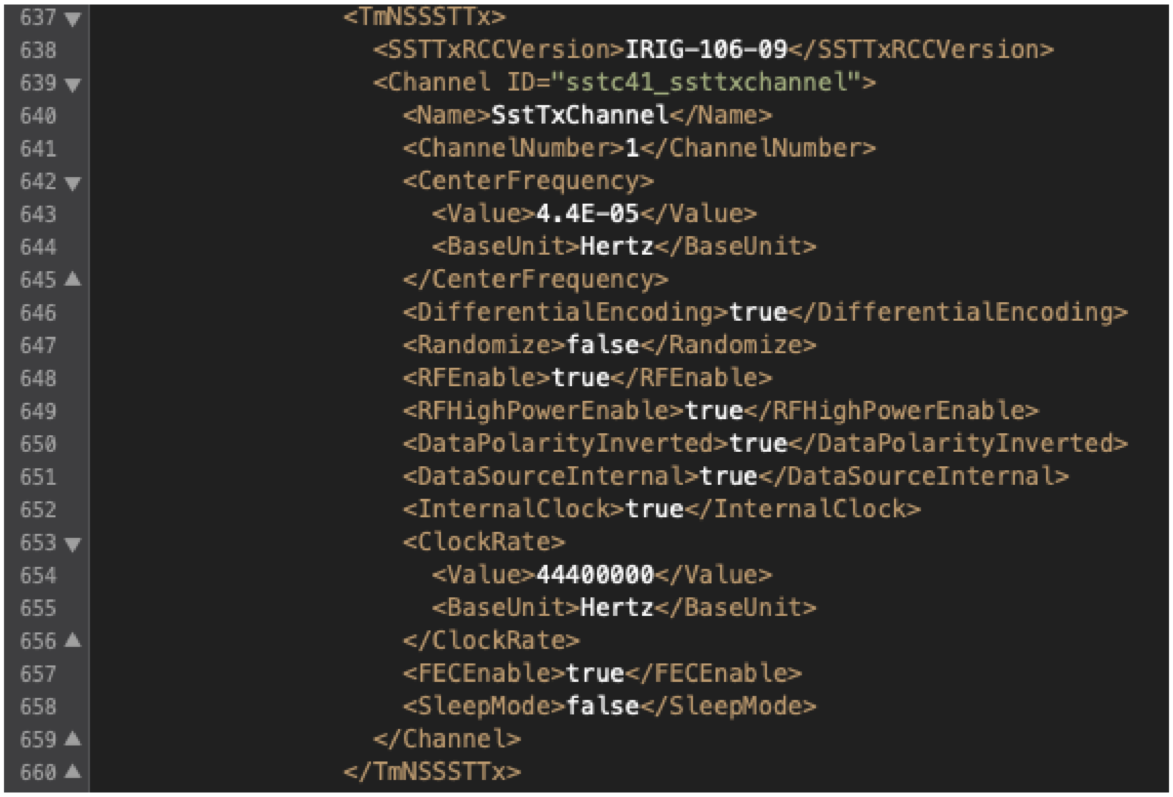
Figure 2-4 shows the MDL configuration elements associated with the TmNSMasterClock capability.



```
<TmNSMasterClock>
  <IEEE1588MasterCapabilityEnable>true</IEEE1588MasterCapabilityEnable>
  <TimeSourceType>ExternalTimeSource</TimeSourceType>
</TmNSMasterClock>
```

Figure 2-4. TmNSMasterClock Capability Parameters

Figure 2-5 shows the MDL configuration elements associated with the TmNSSSTTx capability.



```
<TmNSSSTTx>
  <SSTTxRCCVersion>IRIG-106-09</SSTTxRCCVersion>
  <Channel ID="sstc41_ssttxchannel">
    <Name>SstTxChannel</Name>
    <ChannelNumber>1</ChannelNumber>
    <CenterFrequency>
      <Value>4.4E-05</Value>
      <BaseUnit>Hertz</BaseUnit>
    </CenterFrequency>
    <DifferentialEncoding>true</DifferentialEncoding>
    <Randomize>false</Randomize>
    <RFEnable>true</RFEnable>
    <RFHighPowerEnable>true</RFHighPowerEnable>
    <DataPolarityInverted>true</DataPolarityInverted>
    <DataSourceInternal>true</DataSourceInternal>
    <InternalClock>true</InternalClock>
    <ClockRate>
      <Value>44400000</Value>
      <BaseUnit>Hertz</BaseUnit>
    </ClockRate>
    <FECEnable>true</FECEnable>
    <SleepMode>false</SleepMode>
  </Channel>
</TmNSSSTTx>
```

Figure 2-5. TmNSSSTTx Capability Parameters

2.3.5 Link Manager

A TmNS LM is a software application that is responsible for dynamic link allocations for a particular RAN frequency. It also supports TSS capabilities, both server and client, which is a capability that supports the various handoff modes (antenna-to-antenna [A2A], network-to-network [N2N], and range-to-range [R2R]). In this handbook, a TmNS LM has the system management capabilities and related MDL elements as found in Table 2-6.

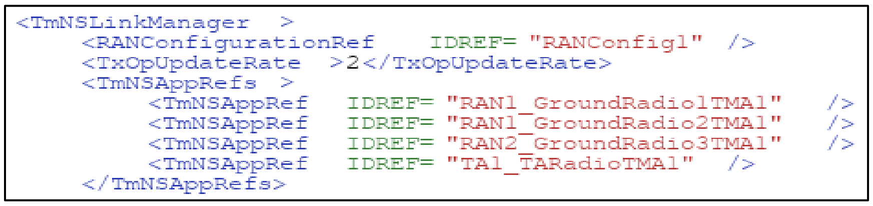
Table 2-6. TmNS Link Manager Capabilities

System Management Resources	MDL Element Tree
tmnsLinkManager	<TmNSLinkManager>
tmnsRAN	<RANConfiguration>
tmnsTmnsSourceSelector	<TmNSTSSClient>

The previously mentioned example MDL file, here, contains a TmNS LM.

The relevant portions of the MDL file associated with the TmNS LM and its capabilities are included in the following pictures.

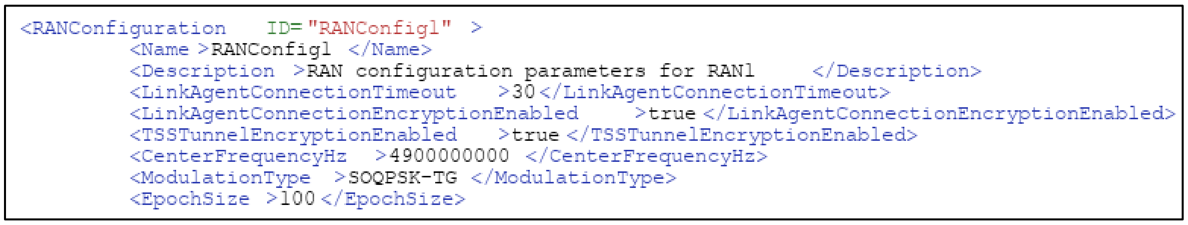
Figure 2-6 shows the MDL configuration elements associated with the TmNSLinkManager capability.



```
<TmNSLinkManager  >
     <RANConfigurationRef    IDREF= "RANConfig1"  />
     <TxOpUpdateRate  >2</TxOpUpdateRate>
     <TmNSAppRefs  >
          <TmNSAppRef   IDREF= "RAN1_GroundRadio1TMA1"   />
          <TmNSAppRef   IDREF= "RAN1_GroundRadio2TMA1"   />
          <TmNSAppRef   IDREF= "RAN2_GroundRadio3TMA1"   />
          <TmNSAppRef   IDREF= "TA1_TARadioTMA1"   />
     </TmNSAppRefs>
```

Figure 2-6. TmNSLinkManager Capability Parameters

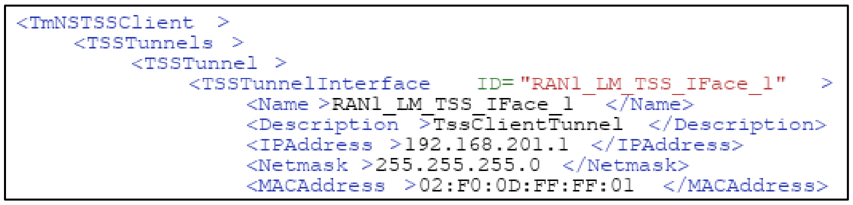
Figure 2-7 shows the MDL configuration elements associated with the RANConfiguration capability.



```
<RANConfiguration   ID= "RANConfig1"  >
        <Name >RANConfig1 </Name>
        <Description >RAN configuration parameters for RAN1    </Description>
        <LinkAgentConnectionTimeout   >30</LinkAgentConnectionTimeout>
        <LinkAgentConnectionEncryptionEnabled    >true</LinkAgentConnectionEncryptionEnabled>
        <TSSTunnelEncryptionEnabled   >true</TSSTunnelEncryptionEnabled>
        <CenterFrequencyHz  >4900000000 </CenterFrequencyHz>
        <ModulationType  >SOQPSK-TG </ModulationType>
        <EpochSize >100</EpochSize>
```

Figure 2-7. RANConfiguration Capability Parameters

Figure 2-8 shows the MDL configuration elements associated with the TmNSTSSClient capability.



```
<TmNSTSSClient  >
    <TSSTunnels >
        <TSSTunnel  >
            <TSSTunnelInterface    ID= "RAN1_LM_TSS_IFace_1"   >
                <Name >RAN1_LM_TSS_IFace_1  </Name>
                <Description >TssClientTunnel  </Description>
                <IPAddress >192.168.201.1 </IPAddress>
                <Netmask >255.255.255.0 </Netmask>
                <MACAddress >02:F0:0D:FF:FF:01  </MACAddress>
```

Figure 2-8. TmNSTSSClient Capability Parameters

2.3.6 Voice Gateway

A TmNS voice gateway component supports VoIP calls over the network. On the surface, this component has a single, fixed function: providing VoIP. However, the capability can be incorporated into other platforms as may be specified by a component specification document. In this handbook, a TmNS voice gateway is a single-function component that has the system management capabilities and related MDL elements as found in Table 2-7.

Table 2-7. TmNS Voice Gateway Capabilities

System Management Resources	MDL Element Tree
tmnsVoiceGateway	<TmNSVoiceGateway>

The previously mentioned example MDL file, here, contains a TmNS voice gateway.

2.3.7 ACU

A TmNS ACU supports the system management interface associated with the tmnsACU management capabilities. These management capabilities provide typical status information from ACUs as well as allowing for a pointing command to be provided. In this handbook, a TmNS ACU is a single-function component that has the system management capabilities and related MDL elements as found in Table 2-8.

Table 2-8. TmNS ACU Capabilities

System Management Resources	MDL Element Tree
tmnsACU	<TmNSACU>

The previously mentioned example MDL file, here, contains a TmNS ACU.

2.3.8 Encryptor

Type 1 network encryptors are likely to be found throughout the system securing the network traffic between TAs and mission control rooms (MCRs). Though there are currently no specific TmNS management variables for network encryptor components within the network, they are worth mentioning as a potential component in the system. The MDL defines minimal description of configuration information for encryptors to define the plaintext (e.g., red enclave) and ciphertext (e.g., black enclave) network interfaces, though these descriptions are provided for system completeness and are not necessarily intended to configure an encryptor. Encryptor configuration is out of scope of the TmNS. In this handbook, network encryptors are treated as a network component with no TmNS-specific interface. The associated MDL elements are found in Table 2-9.

Table 2-9. TmNS Encryptor Capabilities

System Management Resources	MDL Element Tree
N/A	<TmNSEncryptor>

The previously mentioned example MDL file, here, contains a TmNS encryptor.

2.3.9 SST Transmitter

A TmNS SST transmitter supports the system management interfaces and the controls associated with the tmnsSSTTx management capabilities along with providing the functionality of an IRIG 106 Chapter 2[5] SST transmitter. The status and control commands defined in Chapter 2 Appendix 2-C have been provided a corresponding management resource within Chapter 25. Thus, the difference between a TmNS SST transmitter and a non-TmNS SST transmitter is that the TmNS component provides the MDL configuration and system management interfaces as defined in Chapter 23 and Chapter 25.

One way that TmNS SST transmitters have been built and tested is by a TmNS radio serving as a TmNS proxy device for a non-TmNS SST transmitter. Under such a configuration, the TmNS radio advertises the TmNS SST transmitter capability and provides the TmNS system management interface for the component. The TmNS radio connects to a Chapter 2 SST transmitter via an RS-422 serial connection. The TmNS radio translates the TmNS management queries and commands to the corresponding serial commands and sends to the SST transmitter. From a TmNS network connectivity and configuration perspective, the TmNS SST transmitter capability is rolled into the TmNS radio component's functionality. The TmNS SST transmitter examples provided in this handbook assume this capability is used in conjunction with the TmNS radio capability within a single TMA on a component.

In this handbook, the TmNS SST transmitter capability is provided in conjunction with the TmNS radio capability on the same hardware platform. The specific TmNS SST transmitter system management capabilities and related MDL elements as found in Table 2-10.

Table 2-10. TmNS SST Transmitter Capabilities

System Management Resources	MDL Element Tree
tmnsSSTTx	<TmNSSSTTx>

The previously mentioned example MDL file, here, contains a TmNS transmitter capability.

2.3.10 System Manager

A TmNS SysMgr is a multi-role software application that is responsible for configuring, controlling, managing, and monitoring the other TmNS components in the system. It provides a graphical user interface (GUI) and is the primary tool a user has for interfacing with the TmNS network system. The SysMgr component is different than the other TmNS components in that it does not provide a system management interface such as an SNMP agent. Instead, it serves as an SNMP client that issues queries and commands to all other components on the network. It also is not configured with an MDL file. On the contrary, it generates the MDL files by which all other components in the system are configured.

The SysMgr can play many roles throughout the life cycle of a test. It is used in the planning stages to identify components, build topologies, and build and assign data messages, which can be done off-network without any other TmNS devices. It is also used to configure, to monitor for status, and to control the TmNS devices during flight test missions as well as any pre-flight and post-flight system operations.

A test range will likely have multiple SysMgr instances in operation. A SysMgr instance may be found in each separate MCR to manage the separate red enclave networks during flight

[5] Range Commanders Council. "Transmitter and Receiver Systems." In *Telemetry Standards*. RCC 106-20 Chapter 2. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

test missions. Another SysMgr instance is expected to be available within the range operations center to monitor the assets that lie within the black enclaves, such as the airborne and ground radios and LMs. Instances of the SysMgr may also be found in the ground support equipment cart for use during pre-flight checkout operations.

The current implementation of the SysMgr application is for the Windows 10 operating system.

2.4 Non-TmNS Components Within a TmNS System

It is worth mentioning that non-TmNS components may exist within a TmNS system. Commercial-off-the-shelf (COTS) devices may be common within the system to perform non-TmNS-specific functions. One example is a network switch with no TmNS-specific capabilities. These switches may make up a portion of the underlying network infrastructure. Non-TmNS components can be utilized within the TmNS system if they do not prevent the system from being able to perform its function. As an example, a non-TmNS network switch that supports IEEE 1588-2008 may be an acceptable choice for use as part of a TA network, but one that does not support IEEE 1588-2008 is not an acceptable option.

The presence of non-TmNS components within some aspects of a TmNS system is possible and likely probable when considering the overall range network infrastructure as part of the network system. When possible and practical, though, a TmNS-compliant component is recommended as these components will contain better management and status visibility by a SysMgr component as well as allow for configuration through the common MDL configuration language. Configuration and management of non-TmNS components is considered out-of-band and is thus not addressed by either the TmNS or this handbook.

This page intentionally left blank.

CHAPTER 3

Data Acquisition Capability

A TmNS system with a data acquisition capability contains DAUs that produce TDMs onto the network and a recorder to receive and store those TDMs on its local storage media. Data processing applications are also indirectly involved as they are needed for post-processing the recorded data.

While it is possible to construct a recorder-less data acquisition system, it would require a telemetry link (TmNS or SST) to get the acquired data to the ground for live processing. Scenarios in later sections of this handbook describe acquired data over telemetry. As such, this chapter will focus on having the recorder onboard the instrumentation network.

A brief overview of TDMs is provided in this chapter in order to understand how TmNS data is transported over the system,. Following the discussion of TDMs, the remainder of this chapter describes the details associated with the TmNS components required to provide the data acquisition capability. The components required include the following.

- DAU
- Recorder
- Network Switch
- Data Processing Applications
- System Manager

An example MDL file for the components listed above is provided in their respective sections below to highlight the relevant MDL elements for those components.

Figure 3-1 provides the minimal hardware configuration for the data acquisition capability of a TA instrumentation network. The data processing applications and the SysMgr application are not included in the figure as their use cases under this scenario are for pre-flight and post-flight usage. There are multiple TAs depicted in the range-level view of Figure 1-1, but only one is shown here in Figure 3-1. For the sake of this chapter, the real difference between the two lies in the switch configuration parameters. Example MDL files will be provided from each one to demonstrate the different configuration options.

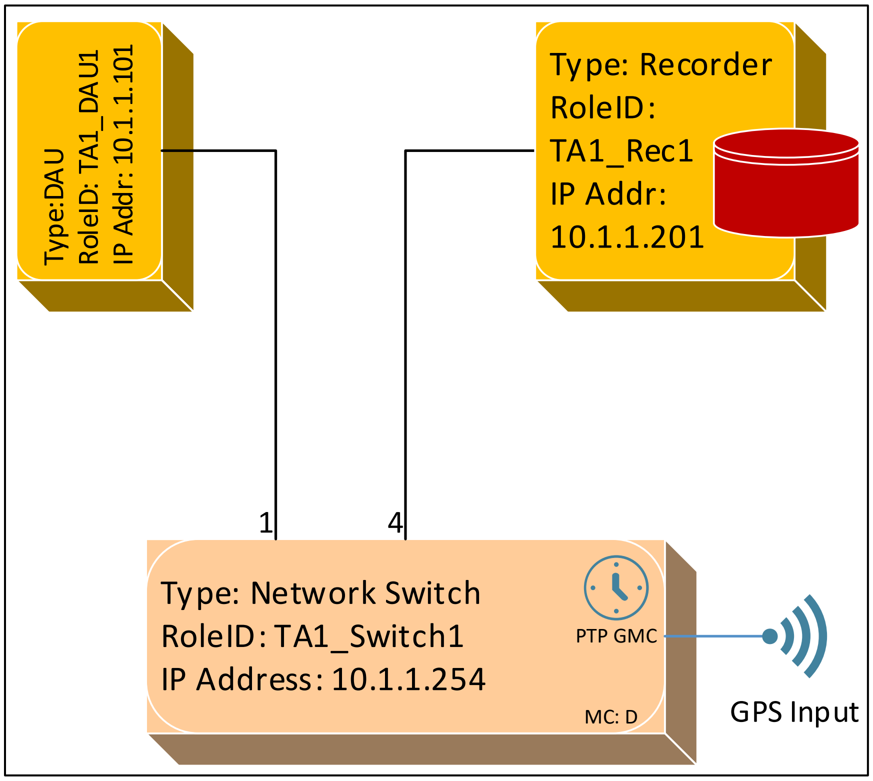


Figure 3-1. Example Components for Data Acquisition Capability on a TA

3.1 TmNS Data Messages

Acquired data is transported over the network via TDMs, which provide the structure of the network packets that carry acquired data from source to sink. This includes data acquired by a DAU and sent to a recorder as well as recorded data being retrieved from a recorder and sent to a data processing application across the network. Chapter 24[6] describes the generic TDM structure.

Three levels comprise TDMs: message-level, package-level, and measurement-level. Each level is discussed in more detail in the following sections. The TDM delivery protocols are also discussed.

3.1.1 Message Structure Details

At the top, message-level, a TDM is comprised of a TmNS message header followed by a TmNS message payload. Chapter 24 provides the message structure, which is illustrated in Figure 3-2.

[6] Range Commanders Council. "Message Formats." In *Telemetry Standards*. RCC 106-20 Chapter 24. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

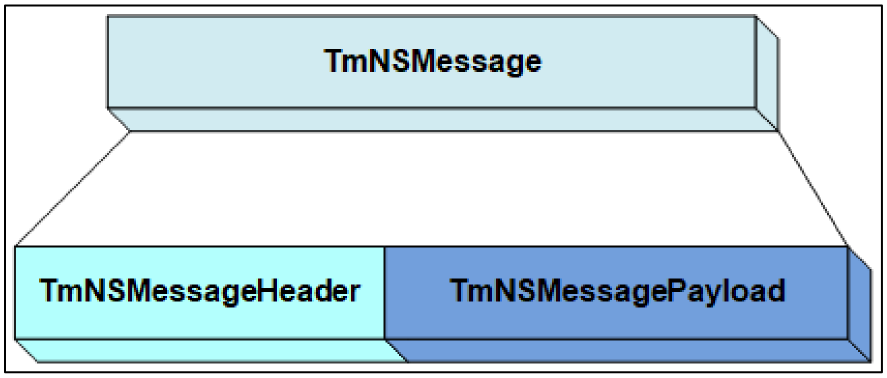


Figure 3-2. TmNS Data Message Structure

The message header provides several fields for identifying a particular message. These fields, their meanings, and their possible values are described in Chapter 24. The TmNS message header is shown in Figure 3-3. Note the figure identifies 32 bits per single row, from 31 down to 0. By the bit numbering convention defined in Chapter 21,[7] the most significant bit of a field is the left-most bit and is assigned the highest bit number. The least significant bit of a field is 0.

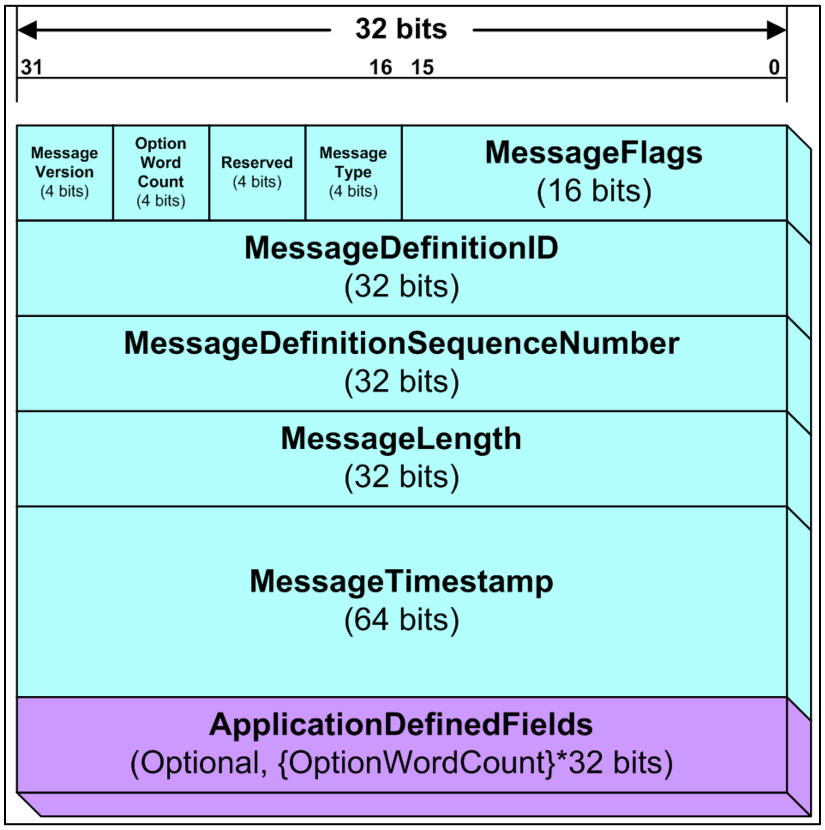

32 bits
31
16 15
0
Message Version (4 bits)
Option Word Count (4 bits)
Reserved (4 bits)
Message Type (4 bits)
MessageFlags (16 bits)
MessageDefinitionID (32 bits)
MessageDefinitionSequenceNumber (32 bits)
MessageLength (32 bits)
MessageTimestamp (64 bits)
ApplicationDefinedFields (Optional, {OptionWordCount}*32 bits)

Figure 3-3. TmNS Data Message Header Structure

[7] Range Commanders Council. "Telemetry Network Standard Introduction." In *Telemetry Standards*. RCC 106-20 Chapter 21. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

As of IRIG 106-19, there has only been one message version and one type of TmNS message defined, the TDM. The Message Definition ID (MDID) is the key identifier for each message in the system. The unique MDID is used to decode the message payload content. The MDL configuration file contains references to any packages that are contained within the message payload. Multiple unique packages per message are possible. Multiple repetitions of the same package are also possible, particularly useful when the time spacing between samples is known.

The sequence number field increments each time a device transmits a new message for the associated MDID. Data sinks, such as recorders, can detect and report lost data whenever the sequence number between received messages for the specific MDID is greater than 1.

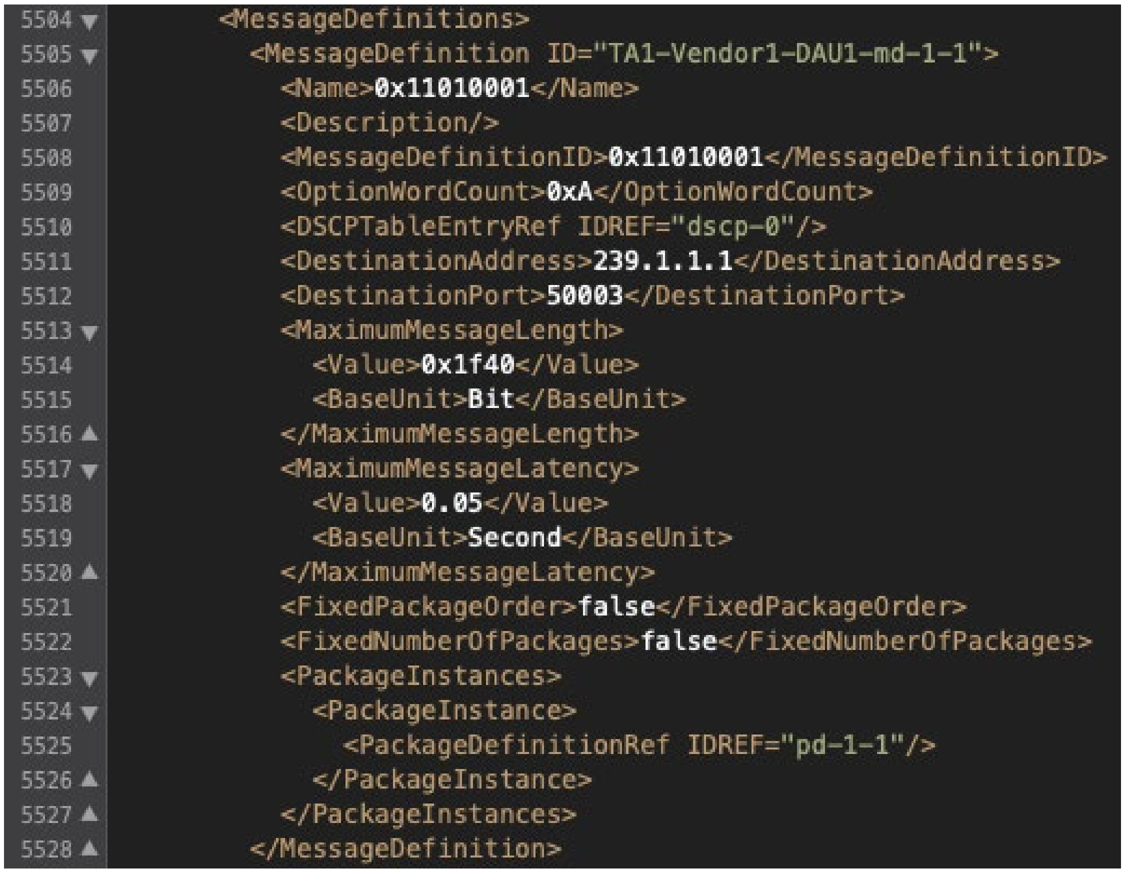
The message length field is used by data processing applications upon receiving a message. The length field allows the application to know how big the message payload is, which in turn provides the application the starting point of the next message received.

Each message header contains a 64-bit timestamp. From this timestamp, the data acquisition time can be determined for each measurement contained within the message payload.

Application-defined fields are optional in a TDM. As such, they may not be supported by all TDM-generating hardware. If they are supported by the source device, they can be enabled to provide additional detail about the TDM within the header. Specific options are defined in Table 24-1 of Chapter 24. These options may prove useful for data processing applications. For instance, an application may find it useful for the IPv4 multicast address to be included within an option of the TDM. An application may also benefit from knowing the number of packages within the TDM. There is also an option for indicating an error condition on the data source. The caveat to using these application-defined fields is that they incur additional network overhead, the impact of which depends on the length of the additional fields and how often they are attached to the TDMs being transmitted over the network.

The MDL description of an MDID also contains transport attributes, such as the destination IP address and port to deliver the packet to. It also contains message delivery constraints such as the maximum allowable latency and maximum message lengths. These constraints are used to strike a proper balance of increased throughput based on large message sizes versus the delivery latency of the messages.

Figure 3-4 shows the definition and description of a single message in the MDL example file. This example contains only a single package definition reference.



```
      <MessageDefinitions>
        <MessageDefinition ID="TA1-Vendor1-DAU1-md-1-1">
          <Name>0x11010001</Name>
          <Description/>
          <MessageDefinitionID>0x11010001</MessageDefinitionID>
          <OptionWordCount>0xA</OptionWordCount>
          <DSCPTableEntryRef IDREF="dscp-0"/>
          <DestinationAddress>239.1.1.1</DestinationAddress>
          <DestinationPort>50003</DestinationPort>
          <MaximumMessageLength>
            <Value>0x1f40</Value>
            <BaseUnit>Bit</BaseUnit>
          </MaximumMessageLength>
          <MaximumMessageLatency>
            <Value>0.05</Value>
            <BaseUnit>Second</BaseUnit>
          </MaximumMessageLatency>
          <FixedPackageOrder>false</FixedPackageOrder>
          <FixedNumberOfPackages>false</FixedNumberOfPackages>
          <PackageInstances>
            <PackageInstance>
              <PackageDefinitionRef IDREF="pd-1-1"/>
            </PackageInstance>
          </PackageInstances>
        </MessageDefinition>
```

Figure 3-4. MDL Portion of the Definition of a TmNS Data Message

3.1.1.1 Message Payload Packing Rules

There are a couple of message configuration options that govern how packages may be packed into the message payload following the message header. The message may have a fixed length, a fixed order of packages within the payload, and/or a fixed number of packages. These configuration parameters are described in the following subsections.

3.1.1.1.1 Fixed Message Length

If a TDM is defined to have a specific, non-varying message size, then the MDL description of this message will contain the <FixedMessageLength> element. This type of message is best suited for message payloads that contain a single package instance or packages that contain periodically sampled measurements. This will ensure that the message length is reached within an expected amount of time. Aperiodic measurements should not be included in messages with a fixed length.

3.1.1.1.2 Fixed Package Order

The <FixedPackageOrder> element is a required configuration parameter for every message defined in an MDL file. This is a true or false flag that indicates whether or not the packages within the message payload are required to maintain a consistent order or not with respect to other packages that may appear in the message payload.

When this value is set to “false”, the order that packages may appear within the message payload is not guaranteed. As such, each package within the message will need to be unambiguously determined. In other words, a data processing application will need to be able to identify each package within the payload correctly. Because the package order is variable, each of the packages in the message should use a common package header with the package identifier, such as the TmNS-defined standard package header.

When this value is set to "true", the order that packages appear within each message is guaranteed. As an example, a message that contains package instances P1, P2, and P3 in that order may produce message payloads that contain any of the following payloads:

- [P1, P2, P3]
- [P1, P2, P3, P1, P2, P3]
- [P1, P2, P3, P1]

Note that in the examples, there may be more than one instance of each package, if the number of packages is not fixed. It is also worth pointing out that the sequence does not need to be complete, as in the last example payload where an extra instance of package P1 was included but not an extra instance of P2 or P3. This still satisfies the constraint in that the package order was preserved.

Having a fixed order of packages removes any ambiguity as to which package comes next, as is the case when the order is not fixed. For this reason, the TmNS-defined standard package header is not required for these messages, though it can still be used. Package header details are discussed further in Subsection 3.1.2.

3.1.1.1.3 Fixed Number of Packages

The <FixedNumberOfPackages> element is a required configuration parameter for every message defined in an MDL file. This is a true or false flag that indicates whether or not the packages within the message payload are required to occur a specific number of times or not.

When this value is set to "true", the packages in the message payload will appear exactly the number of times as they appear in within the MDID description. Multiple occurrences of the same package are possible by specifying the same package reference multiple times. When using a fixed number of packages per message, the message will likely be set to a fixed message length.

When this value is set to "false", packages associated with the message may appear any number of times within the payload, including zero times. When a fixed number of packages per message is not being used, the message is not a fixed-length message and will use other transmission frequency condition parameters to determine when to send the TDM with a variable number of packages.

When multiple instances of the same package are provided within the same message instance, the measurement values contained by those package instances may be different. The time between those samples may be communicated within a package header field, such as the TmNS-defined standard package header's "PackageTimeDelta" field, or it may be understood through the MDL description of the particular package definition.

3.1.1.2 Message Transmission Frequency Conditions

The frequency of how often a message is sent from a DAU may be determined by more than just how often the DAU samples data. When a message does not contain the <FixedMessageLength> element, then its actual length is variable. Thus, one instance of an

MDID may be 1,000 bytes long, and the next instance of the same MDID may only be 800 bytes long.

A TDM that is of variable length rather than a fixed size may be a desired setting for messages that transport measurements that are not necessarily provided at fixed rates and known intervals. For variable length messages, there are message configuration parameters that provide thresholds that relate to both maximum length and latency. Messages are sent such that both of these configuration parameters are met. Though not required to be used together, the maximum length and maximum latency thresholds are most often used together. The TDM is generated once the first threshold is met, whether it be the maximum message length or the maximum message latency. These two thresholds are further described in the following subsections.

3.1.1.2.1 Maximum Message Length

The <MaximumMessageLength> configuration element indicates the maximum message length of the TDM. As a DAU or other TDM source builds a message, it may continue to add additional packages into the payload until the message length reaches the specified maximum message length or until an additional package would cause the message length to exceed the specified maximum message length. When either of these cases are encountered, the payload is finalized, and the TDM can then be sent.

3.1.1.2.2 Maximum Message Latency

The <MaximumMessageLatency> configuration element indicates the maximum amount of time that a DAU or TDM source can buffer acquired data before sending it over the network with a TDM. When a measurement is acquired, it is packaged, and the package is then presented to the device’s TDM builder process to be placed in the message payload. Waiting to send the TDM allows for additional packages to be added to the TDM payload. The maximum message latency provides the upper limit on how long to wait before sending the TDM.

Transmitting larger payloads improves the overall network efficiency in that it allows sending fewer larger packets instead of smaller packets more often, which ultimately cuts down on some of the network overhead associated with transmitting each packet. The maximum message latency parameter is meant to allow message sizes to keep growing until the maximum latency threshold is reached. Once this threshold is reached, the TDM is finalized and sent onto the network. This latency threshold is used to ensure that any time-critical data, such as safety of flight data, is not buffered too long on the acquisition device before transmitting.

3.1.2 Package Structure Details

Within the payload of a TDM, there is one or more packages. These packages may be unique packages, or they may be multiple instances of the same package, likely carrying different samples of the same measurement. Like the message structure, a package also has its own structure. Chapter 24 defines the basic structures, which can be seen in Figure 3-5.

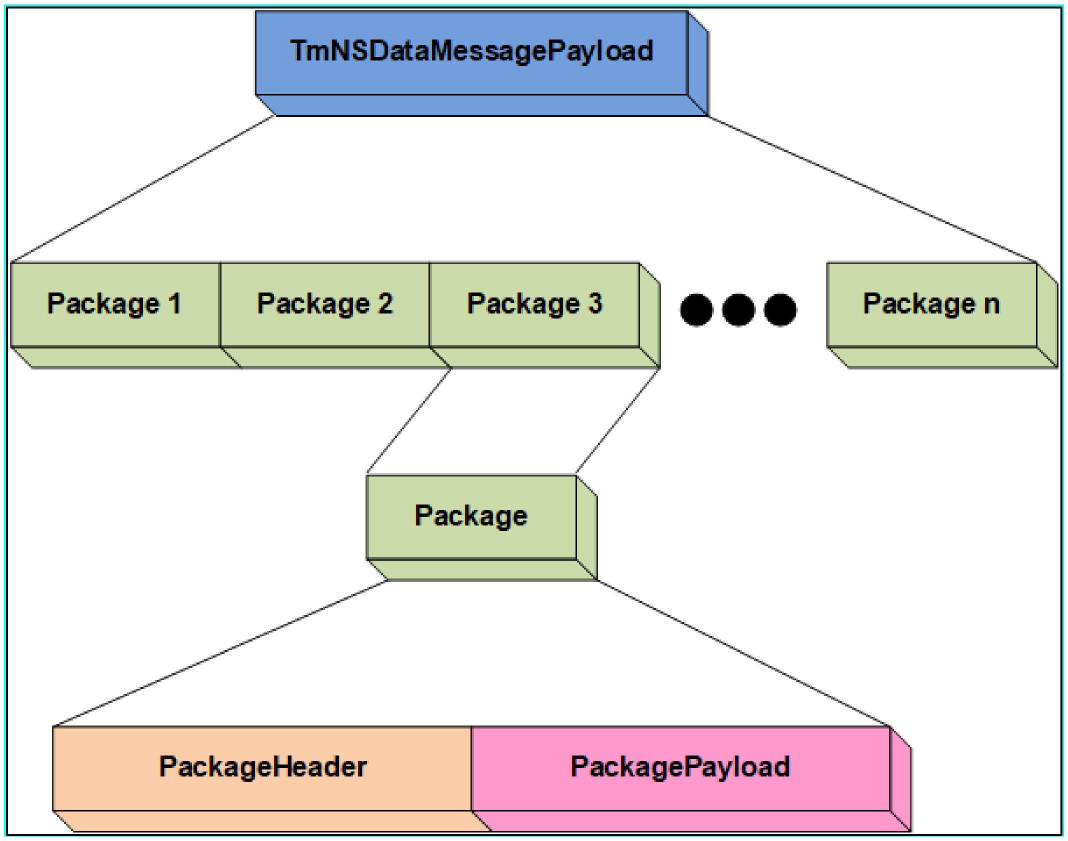


Figure 3-5. TDM Payload with Package Structure

A TDM payload may contain multiple packages, and these packages may consist of multiple unique packages or multiple instances of the same package. Each package also has a defined structure. The general structure is similar to that of a message, containing a package header followed by the package payload. However, the structure of the package header is not guaranteed.

The TmNS provides a standard package header definition. When the standard header is used on the package, the package can be identified in similar fashion as the message is identified. The standard package header contains fixed fields for a Package Definition ID (PDID), the length of the package, package status flags, and a time delta field. The PDID is the key identifier for each package in the system. The unique PDID is used to decode the specific measurement or measurements contained within the package. The standard package header is depicted in Figure 3-6.

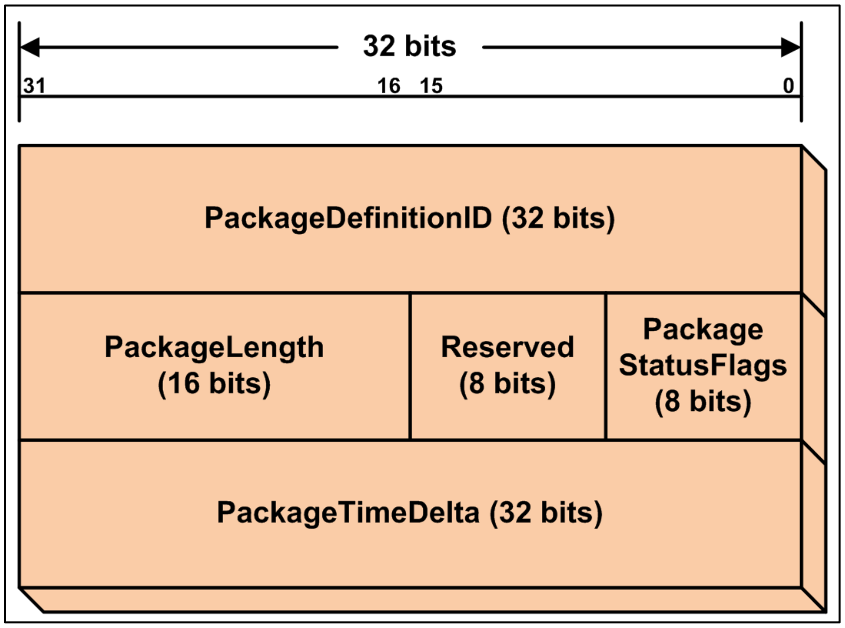
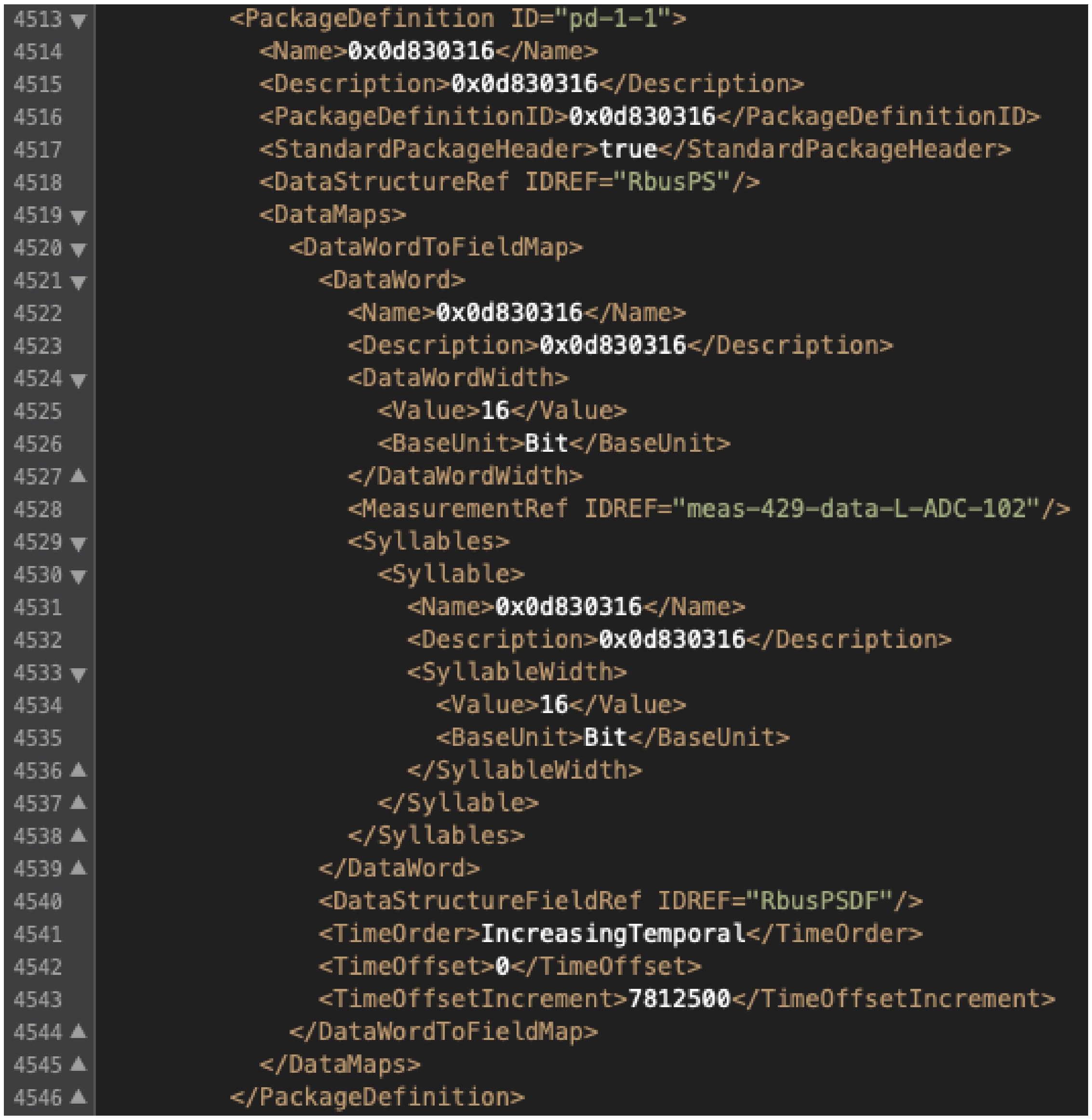

32 bits
31
16
15
0
PackageDefinitionID (32 bits)
PackageLength (16 bits)
Reserved (8 bits)
Package StatusFlags (8 bits)
PackageTimeDelta (32 bits)

Figure 3-6. TmNS-Defined Standard Package Header

The data structure mapping and specific measurement placement within the structure are specified as part of each package definition. Figure 3-7 shows the definition and description of a single package in the MDL example file. In this example, the package defined uses the standard package header.



```xml
<PackageDefinition ID="pd-1-1">
  <Name>0x0d830316</Name>
  <Description>0x0d830316</Description>
  <PackageDefinitionID>0x0d830316</PackageDefinitionID>
  <StandardPackageHeader>true</StandardPackageHeader>
  <DataStructureRef IDREF="RbusPS"/>
  <DataMaps>
    <DataWordToFieldMap>
      <DataWord>
        <Name>0x0d830316</Name>
        <Description>0x0d830316</Description>
        <DataWordWidth>
          <Value>16</Value>
          <BaseUnit>Bit</BaseUnit>
        </DataWordWidth>
        <MeasurementRef IDREF="meas-429-data-L-ADC-102"/>
        <Syllables>
          <Syllable>
            <Name>0x0d830316</Name>
            <Description>0x0d830316</Description>
            <SyllableWidth>
              <Value>16</Value>
              <BaseUnit>Bit</BaseUnit>
            </SyllableWidth>
          </Syllable>
        </Syllables>
      </DataWord>
      <DataStructureFieldRef IDREF="RbusPSDF"/>
      <TimeOrder>IncreasingTemporal</TimeOrder>
      <TimeOffset>0</TimeOffset>
      <TimeOffsetIncrement>7812500</TimeOffsetIncrement>
    </DataWordToFieldMap>
  </DataMaps>
</PackageDefinition>
```

Figure 3-7.   MDL Portion of the Definition of a Package

It is also possible to utilize a custom package header. In this case, the <StandardPackageHeader> field would be set to “false”, and the custom header would be completely defined in the MDL. This also includes the ability to have no package header at all. In either case, whether a custom header is used or none at all, the PDID must be well understood based on the MDID alone. For example, a message containing packages with non-standard or no header must adhere to a strict ordering of packages in order for data processing applications to know how to retrieve and interpret the measurement data contained within the package.

The decision for utilizing standard package headers or not comes down to a tradeoff in the system. With each header, the network overhead increases. This becomes increasingly important to consider when sending TDMs over the constrained RF link. However, by removing the use of standard package headers, there may be an increase in the processing burden on data consumers such as data processing applications. With standard package headers, parsing the message and the packages can utilize a simpler parsing algorithm, but without the standard package headers used throughout all packages in a message, the parsing algorithm must have an additional level of MDL-awareness in its parsing capability. As a guiding rule, the package should default to using the standard package header unless there is sufficient justification for deviating from using it.

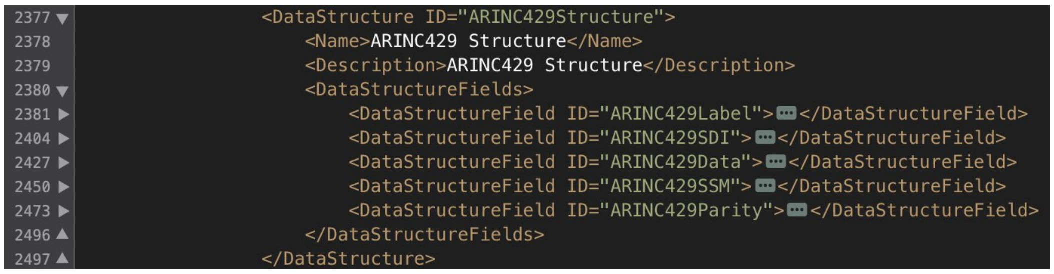
3.1.2.1 Data Structures

Data structures are the main way that MDL describes the various shapes of network data. There are two main places they are used - to define the shape of TmNS packages within PackageDefinitions, and to define the shape of bus messages within DataStreamMessages. The DataStructure MDL definition is designed to be generic and not rely on any assumptions for particular formats. This means that, while there is no built-in structure for the more common data types, any data shape that can be described by a DataStructure can be processed.

Note that much of these structures were fundamentally changed between MDL v1.0.0 and MDL v1.1.0 to address some known shortcomings. These sections were originally written describing MDL v1.0.0, to match current hardware capabilities, but the decision was since made to reflect the newest version of the standards. As such, the updated information to reflect this current version is still being documented.

Note: MDL v1.0.0 Description

Figure 3-8 provides the MDL description of a Data Structure. The example shown is that of an ARINC 429 structure. DataStructures are made of DataStructureFields and DataStructureFieldSets (which in turn are made up of DataStructureFields). Each field has a location and size that describe where to find the field and what the digital properties of it are.



```
<DataStructure ID="ARINC429Structure">
    <Name>ARINC429 Structure</Name>
    <Description>ARINC429 Structure</Description>
    <DataStructureFields>
        <DataStructureField ID="ARINC429Label">⋯</DataStructureField>
        <DataStructureField ID="ARINC429SDI">⋯</DataStructureField>
        <DataStructureField ID="ARINC429Data">⋯</DataStructureField>
        <DataStructureField ID="ARINC429SSM">⋯</DataStructureField>
        <DataStructureField ID="ARINC429Parity">⋯</DataStructureField>
    </DataStructureFields>
</DataStructure>
```

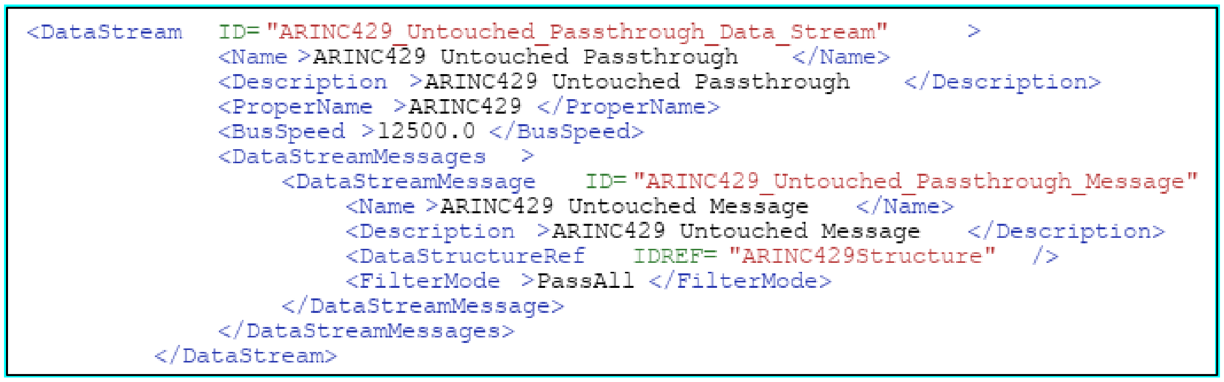
Figure 3-8. MDL Portion of the Definition of a Data Structure

3.1.2.2 Data Streams

Data streams are what MDL uses to describe bus messages. A data stream has one or more DataStreamMessages, each representing one shape of data that may appear on the bus by referencing a DataStructure. DataStreamMessages in turn reference DataStructures, which define what the specifics of that data shape are. This can be anywhere from an ARINC 429 message, to one type of MIL-STD-1553 message, to arbitrary Ethernet data.

The key piece is that data streams are generally used to describe the shapes of data that are acquired. This is in contrast to packages and messages, which are used to describe data that leaves the acquisition hardware. These, combined with the capture capabilities of the DAU (described later), determine what data can be extracted from bus messages, and what data is packaged and put on the network for recording and any further processing. Depending on the use case, this may be used again by data processing to pick apart what the DAU captured and fully extract the desired information.

Figure 3-9 provides an MDL definition of a data stream.



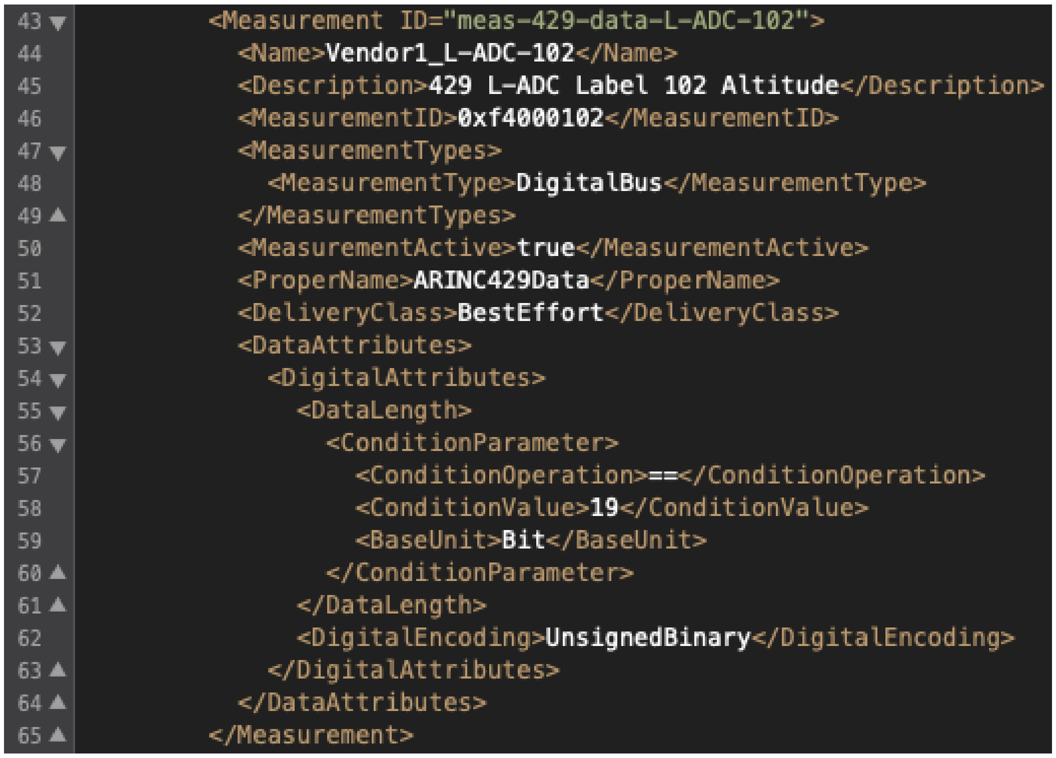
```
<DataStream   ID= "ARINC429_Untouched_Passthrough_Data_Stream"      >
             <Name >ARINC429 Untouched Passthrough    </Name>
             <Description  >ARINC429 Untouched Passthrough    </Description>
             <ProperName  >ARINC429 </ProperName>
             <BusSpeed >12500.0 </BusSpeed>
             <DataStreamMessages   >
                 <DataStreamMessage    ID= "ARINC429_Untouched_Passthrough_Message"
                     <Name >ARINC429 Untouched Message    </Name>
                     <Description  >ARINC429 Untouched Message    </Description>
                     <DataStructureRef    IDREF= "ARINC429Structure"   />
                     <FilterMode  >PassAll </FilterMode>
                 </DataStreamMessage>
             </DataStreamMessages>
         </DataStream>
```

Figure 3-9. MDL Portion of the Definition of a Data Stream

3.1.3 Measurement Details

Measurements are at the heart of what really matters most – the data. Measurement values may consist of counts, engineering units, instrumentation units, other raw readings, or computed values. They often originate from analog samples or digital avionics busses. The MDL file contains measurement descriptions that include a unique Measurement ID to identify the measurement and any data attributes associated with the measurement, such as the length of the data field and the encoding associated with it.

Figure 3-10 shows the definition and description of a measurement in the MDL example file.



```
<Measurement ID="meas-429-data-L-ADC-102">
  <Name>Vendor1_L-ADC-102</Name>
  <Description>429 L-ADC Label 102 Altitude</Description>
  <MeasurementID>0xf4000102</MeasurementID>
  <MeasurementTypes>
    <MeasurementType>DigitalBus</MeasurementType>
  </MeasurementTypes>
  <MeasurementActive>true</MeasurementActive>
  <ProperName>ARINC429Data</ProperName>
  <DeliveryClass>BestEffort</DeliveryClass>
  <DataAttributes>
    <DigitalAttributes>
      <DataLength>
        <ConditionParameter>
          <ConditionOperation>==</ConditionOperation>
          <ConditionValue>19</ConditionValue>
          <BaseUnit>Bit</BaseUnit>
        </ConditionParameter>
      </DataLength>
      <DigitalEncoding>UnsignedBinary</DigitalEncoding>
    </DigitalAttributes>
  </DataAttributes>
</Measurement>
```

Figure 3-10. MDL Portion of the Definition of a Measurement

Note that the definition of Measurement is potentially broader than the traditional interpretation. Measurements are always data, but in addition to the typical analog and digital acquisition sources, some measurements can be the result of data processing or other intermediate results. For example, TmNS supports concepts for not-yet-implemented hardware, such as on-aircraft computational elements, and those devices would still need to move data around. Rather than defining a parallel but functionally equivalent concept just for these computed values, these other pieces of data are still referred to as measurements.

Measurements are contained within packages of TMDs. Their placement within an associated package is formatted according to the data structure or data stream associated with the package. The placement is specified in the package definition.

Additional details of describing and packaging measurements will be discussed under Section 3.2 regarding data acquisition by the DAU.

3.1.4 Transport Protocols

There are two data delivery protocols defined in the TmNS for delivering TDMs. These protocols are found in Chapter 26.[8] The delivery protocols have different transport characteristics in order to satisfy different use cases.

Generally speaking, TDMs being generated by a DAU will utilize the LTC delivery protocol that is a metadata-defined delivery approach where data is delivered upon configuration, if enabled for transmitting. Network delivery takes the form of User Datagram Protocol (UDP) datagrams in IP packets. Though there is no two-way handshaking with UDP to guarantee in-order assured delivery, it is able to provide a lower latency for end-to-end transmission than Transmission Control Protocol (TCP)-based delivery protocols. For this reason, the UDP-based LTC delivery protocol is most often associated with TDM delivery from DAUs utilizing IP multicast.

The other TDM delivery protocol is the RC delivery protocol, and it is most often associated with retrieving data from a TmNS recorder. This is a request-defined data delivery approach. Data is only delivered upon request. Network delivery utilizes TCP, a connection-based protocol that guarantees an in-order delivery as long as the connection is maintained. The protocol will send retry messages if a packet is lost in transit. Retry events slow down the overall transport process, which makes the RC protocol not desirable for live data collection and analysis. However, it is ideal for on-demand data retrieval from a recorder that has already stored the data.

Both the LTC and RC delivery protocols send TDMs. The transport characteristics between the two protocols are what make the difference in delivery performance, where each protocol is used in support of different data transport scenarios.

3.2 DAU

The DAUs acquire the sensor and bus data and place the data onto the network for consumption by end devices (e.g., data recorders, data processing applications, etc.). A TmNS

[8] Range Commanders Council. "TmNSDataMessage Transfer Protocol." In *Telemetry Standards*. RCC 106-20 Chapter 26. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

DAU has at least one network interface over which it produces/publishes TDMs. It also responds to system management queries and commands.

The DAUs may acquire analog or digital data, or both. The DAU places acquired measurements into data *Packages*, and then those *Packages* are grouped into *Messages*. The *Messages* are delivered over the network as a TDM. This section walks the user through setting up acquisition and publishing characteristics for the DAU.

3.2.1 DAU Hardware Topology Concepts

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

This section will describe the typical pieces of a DAU. Though vendors will likely have different architectures to their DAUs, some fixed chassis and others with expandable module stacks, the MDL describes both approaches in a consistent way. With respect to hardware and MDL descriptions, there are big modules/slices/boards that may contain submodules/daughter cards/shields. The MDL can describe the physical connectors on the hardware as well as the logical groupings of pins into ports to which transducers can be connected to.

3.2.2 Analog Acquisition

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

Analog acquisition is the gathering of data through physical transducers. Examples are accelerometers, thermocouples, and strain gauges. Representations of the physical phenomena are then acquired by a DAU and turned into measurements to be put onto the network.

This section will describe the relationships of the transducers and their connection to DAU modules. It also provides examples of how measurements are described. This includes analog and discrete measurements as well as special cases such as time measurements, simultaneously sampled measurements, and storage of calibration information.

3.2.3 Digital Acquisition

Digital acquisition is the gathering of data through existing digital bus networks on the TA. Example sources that could be tapped into are ARINC 429 and MIL-STD-1553 buses. In TmNS, these are described using data streams. This also includes capturing of general Ethernet traffic.

This section will describe how data streams can be set up and transmitted onto the network as TDMs. It will include how to make sure the user only requests captures that fall within the DAU's capturing capabilities.

NOTE	Initial handbook content for this section was written using MDL v1.0.0 descriptions. This was done to align with currently existing hardware. During the course of handbook development, it was decided that the handbook should address the most current version of the standards, MDL v1.1.0. Updated documentation to this section is forthcoming. The original content of this section (and subsections) is included below in this table. Due to the MDL changes

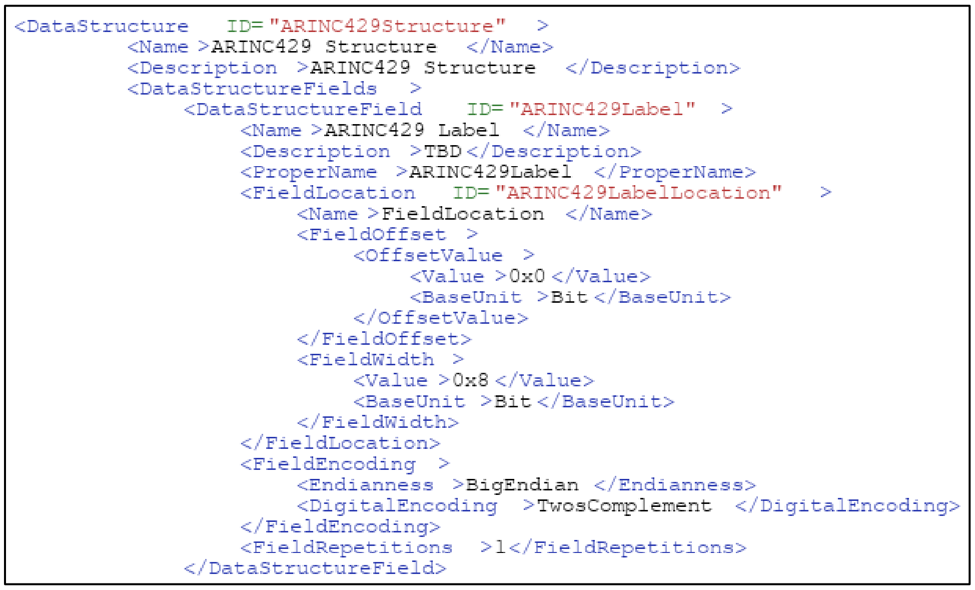
	between v1.0.0 and v1.1.0 with respect to data structures and other packaging capability descriptions, the former documentation could be potentially misleading. It should be removed and replaced with the v1.1.0 documentation when available.

3.2.3.1 Capture Capabilities and Packaging

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

An important thing for a user to keep in mind when they are describing their digital acquisition is the capabilities of their particular capture method. Proper use of the CaptureCapability elements in MDL will prevent requiring a user to peruse datasheets and documentation looking for how to successfully configure a device. There are many different sets of hardware capabilities out there, working at full message level down to individual bits, and it is critical for a user to know how they should set up their acquisition to fit within their particular device's constraints and minimize the chance for miscommunication, or worse, setting up an acquisition scheme that cannot be done by their device.

For this first look at the basic concepts, the ARINC 429 digital bus will be used as the notional example. Messages on this bus follow a fixed-length, 32-bit data structure, making it one of the simpler formats likely to be acquired. By using this as the starting point, the different capture concepts can be explained without getting into some of the complexity involved with incremental decoding and other lengths. Similar concepts will hold true for more complicated acquisition targets, such as MIL-STD-1553 and even generic Ethernet, both of which will be discussed later. Holding off on this additional complexity when explaining the core concepts also has the advantage that this knowledge will not be affected by upcoming changes to the MDL description of these data structures. Figure 3-11 shows the beginning of the MDL description of an ARINC 429 data structure.



```
<DataStructure  ID="ARINC429Structure"  >
        <Name >ARINC429 Structure  </Name>
        <Description >ARINC429 Structure  </Description>
        <DataStructureFields  >
            <DataStructureField  ID="ARINC429Label"  >
                <Name >ARINC429 Label  </Name>
                <Description  >TBD</Description>
                <ProperName  >ARINC429Label  </ProperName>
                <FieldLocation  ID="ARINC429LabelLocation"  >
                    <Name >FieldLocation  </Name>
                    <FieldOffset  >
                        <OffsetValue  >
                            <Value >0x0</Value>
                            <BaseUnit  >Bit</BaseUnit>
                        </OffsetValue>
                    </FieldOffset>
                    <FieldWidth  >
                        <Value >0x8</Value>
                        <BaseUnit  >Bit</BaseUnit>
                    </FieldWidth>
                </FieldLocation>
                <FieldEncoding  >
                    <Endianness  >BigEndian </Endianness>
                    <DigitalEncoding  >TwosComplement  </DigitalEncoding>
                </FieldEncoding>
                <FieldRepetitions  >1</FieldRepetitions>
            </DataStructureField>
```

Figure 3-11. MDL Example Description of an ARINC 429 Data Structure

The SysMgr application provides a visual representation of data structures. Its depiction of the ARINC 429 data structure is displayed in Figure 3-12.

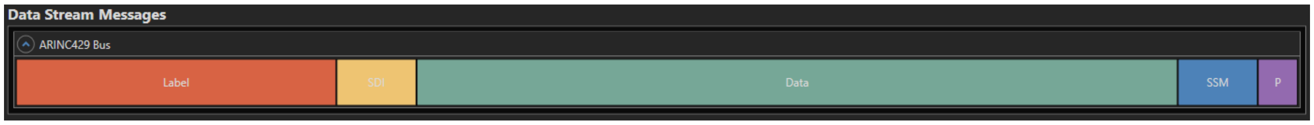


Figure 3-12. ARINC 429 Data Structure

The description of the ARINC 429 data structure is given using the same exact MDL structure as what was previously looked at for packages: the DataStructure. Thus, the details of the description will be glossed over here, and the structure will just briefly be mentioned, as shown in the figure. First, an 8-bit Label, then a 2-bit Source/Destination Identifier (SDI), a 19-bit Data field, another 2-bit field for the Sign/Status Matrix (SSM), and finally 1 bit for parity. These fields will be referenced by name as we move through the different concepts.

Within each of these different capture schemes, first the CaptureCapability will be shown and discussed, which describes what a DAU with that capability can do. Then, a DataStream working within those capabilities will be shown and discussed. Note that this section will not be discussing how to take the DataStreams or extracted measurements and put them into packages and send them out of the DAU, as that was previously described in Subsection 3.1.2.

3.2.3.1.1 Untouched Passthrough

The simplest method of capture is a full-bus capture with no processing or filtering. The DAU simply listens to the bus, and every message that is received is captured for sending on the TmNS network. The DAU looks at no fields for decision-making, and no down-select or field extraction occurs at the site of capture. Given the unprocessed state of the data at this point, it is likely that post-processing will have to occur to extract value from this data later in the process.

Figure 3-13 provides an example of the MDL description of an untouched passthrough message for the full bus capture of an ARINC 429 avionics bus using a data structure in the package.

```
<CaptureCapability   ID= "Passthrough"  >
              <Name >assthrough </Name>
              <Description  >Passthrough of ARINC 429 without filtering or choices      </Description>
              <ProperName  >ARINC429 </ProperName>
              <DataStructureRef   IDREF= "ARINC429Structure"   />
              <CaptureMode  >AllBusCapture  </CaptureMode>
              <PackagingSchemes   >
                   <PackagingScheme   >
                        <Name >Streaming 429 Packaging   </Name>
                        <Description  >ARINC 429 Packaging which takes all fields     </Description>
                        <Exactly >
                             <DataStructureFieldRef    IDREF= "ARINC429Label"  />
                             <DataStructureFieldRef    IDREF= "ARINC429SDI"  />
                             <DataStructureFieldRef    IDREF= "ARINC429Data"  />
                             <DataStructureFieldRef    IDREF= "ARINC429SSM"  />
                             <DataStructureFieldRef    IDREF= "ARINC429Parity"   />
                        </Exactly>
                   </PackagingScheme>
              </PackagingSchemes>
         </CaptureCapability>
```

Figure 3-13. MDL Example Description of an Untouched Passthrough for Full Bus Capture

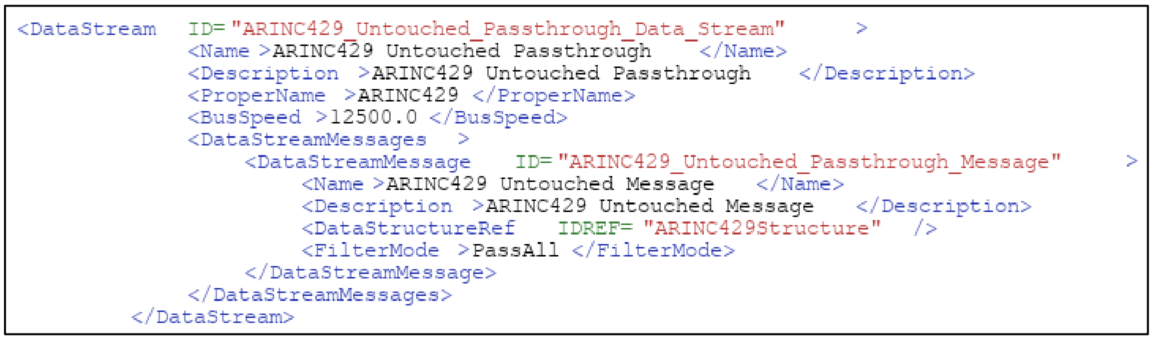
The first important elements in this portion of MDL are the ProperName and DataStructureRef. Together, these tell what types of digital bus data the capability is applicable to. In this example, we are looking at ARINC 429 messages, so the ProperName is ARINC429 and the DataStructureRef points to the DataStructure we showed above. The DataStructure is particularly important because it determines where the fields that can be used for filtering and extraction come from. In an upcoming version of the MDL standard, this has changed, but for the purposes of this handbook, multiple DataStructures will likely be required to describe the different message shapes that may appear on a digital bus with more variable message shapes. This plays into the choice to use ARINC 429 as this first teaching example.

The next meaningful element is the CaptureMode. The value here is AllBusCapture, which means that this particular capability does not have any filtering and will take everything on the bus along to the next step, which lines up with the current goal.

Finally, the last meaningful piece in this example is the PackagingScheme. There are several options that will slowly be uncovered over the course of this section, but for now, the scheme shown is for all of the ARINC 429 fields. This means that there is no opportunity to drop or selectively choose what data fields are acquired and that the only option is to take all of them.

In summary, this capability states that the DAU can capture from an ARINC 429 bus without any filtering and without any opportunity to drop data. Next, a possible DataStream will be defined that follows the rules set out by this CaptureCapability.

Figure 3-14 provides an MDL description of a DataStream for ARINC 429.



```
<DataStream   ID="ARINC429_Untouched_Passthrough_Data_Stream"     >
              <Name >ARINC429 Untouched Passthrough    </Name>
              <Description >ARINC429 Untouched Passthrough    </Description>
              <ProperName >ARINC429 </ProperName>
              <BusSpeed >12500.0 </BusSpeed>
              <DataStreamMessages  >
                  <DataStreamMessage   ID="ARINC429_Untouched_Passthrough_Message"     >
                      <Name >ARINC429 Untouched Message    </Name>
                      <Description >ARINC429 Untouched Message    </Description>
                      <DataStructureRef   IDREF="ARINC429Structure"   />
                      <FilterMode >PassAll </FilterMode>
                  </DataStreamMessage>
              </DataStreamMessages>
          </DataStream>
```

Figure 3-14. MDL Example Description of an Untouched Passthrough with a DataStream

The DataStream in Figure 3-14 also describes an untouched passthrough. There is only one DataStreamMessage, since there is only one data shape that could ever be present on the ARINC 429 bus. The DataStructureRef points to the ARINC429Structure, which is the same that is used by the CaptureCapability. Finally, the FilterMode is set to PassAll, which means that the DAU should capture all messages unless otherwise defined. There are no exceptions defined, which indicates that the DAU is capturing everything, which fits within this DAU's capabilities.

As shown in Subsection 3.1.2, a package that is defined to use a data steam rather than a data structure would then reference the DataStreamMessage directly, providing no opportunity for modification or reorganization.

3.2.3.1.2 Reorganized Passthrough

A reorganized passthrough, also referred to as "streaming through measurements", arises from a similar situation. No filtering or data selection will be done on the DAU, but the data that is output may not be in exactly the same shape as how it was originally captured. A classic example of this is what could occur when loading ARINC 429 messages into a PCM stream, where the synchronous nature of the stream may necessitate non-contiguous fields, such as splitting up the label from the rest of the ARINC 429 message.

The CaptureCapability is the same in this case. Again, the DAU cannot be asked to filter on anything or to make any choices about the data based on the values within the fields. Instead, the difference comes in the description of the DataStream and the data, and the other supporting information that is given by the DAU. The ability to do a reorganized passthrough is indicated by the DAU supplying measurements with ProperNames matching each field, in addition to such a CaptureCapability. An example of this is shown in Figure 3-15.

```
<Measurement   ID="ARINC429_Data_Measurement"    >
    <Name >ARINC429 Data  </Name>
    <Description  >Data </Description>
    <MeasurementID  >0xF0000001  </MeasurementID>
    <MeasurementTypes   >
        <MeasurementType  >DigitalBus  </MeasurementType>
    </MeasurementTypes>
    <MeasurementActive  >false </MeasurementActive>
    <ProperName  >ARINC429Data  </ProperName>
    <DeliveryClass  >BestEffort  </DeliveryClass>
</Measurement>
```

Figure 3-15. MDL Example Description of a Reorganized Passthrough

Given a set of measurements to stream through, and that CaptureCapability, the DataStream would now need to include a new element: the MeasurementSelectors, shown in Figure 3-16.

```
<MeasurementSelectors   >
    <MeasurementSelector   >
        <FieldAssignments   >
            <FieldAssignment   >
                <DataStructureFieldRef     IDREF= "ARINC429Label"  />
                <MeasurementRef    IDREF= "ARINC429_Label_Measurement"    />
            </FieldAssignment>
            <FieldAssignment   >
                <DataStructureFieldRef     IDREF= "ARINC429SDI"  />
                <MeasurementRef    IDREF= "ARINC429_SDI_Measurement"    />
            </FieldAssignment>
            <FieldAssignment   >
                <DataStructureFieldRef     IDREF= "ARINC429Data"  />
                <MeasurementRef    IDREF= "ARINC429_Data_Measurement"    />
            </FieldAssignment>
            <FieldAssignment   >
                <DataStructureFieldRef     IDREF= "ARINC429SSM"  />
                <MeasurementRef    IDREF= "ARINC429_SSM_Measurement"    />
            </FieldAssignment>
            <FieldAssignment   >
                <DataStructureFieldRef     IDREF= "ARINC429Parity"  />
                <MeasurementRef    IDREF= "ARINC429_Parity_Measurement"    />
            </FieldAssignment>
        </FieldAssignments>
```

Figure 3-16. MDL Example Description of a Measurement Selector for Reorganized Passthrough

Details of this element will be expanded as more complicated configurations are requested, but for now, there is just one MeasurementSelector, which means it applies to every incoming message. There are then five FieldAssignments, each of which takes an ARINC429 field and assigns it to the corresponding ARINC 429 measurement. Thus, after capture, we are left with a measurement per field, each of which gets a new value as a new ARINC 429 message is acquired.

Packaging then proceeds as normal, using DataWordToFieldMaps to assign these measurements into fields in an outgoing package, with one major exception: the CaptureCapability declared that `<Exactly>` all of the fields must be used in its PackagingScheme. Therefore, for any package to be valid, it must contain exactly all of the measurement fields. The difference with doing it with this method versus the untouched passthrough way is that the fields do not necessarily have to be in order or make up the exclusive content of the package.

3.2.3.1.3 Measurement Extraction

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

This section provides an MDL description of measurement extraction. It describes how acquired data is pulled from specific fields into measurements, which allows those measurements to be packaged directly.

3.2.3.1.4 Filters vs. Extraction

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

This section provides a description of the difference between filtering the data and extracting it. Filters are used to prevent what data even makes it into the DAU for internal processing (e.g., we will only look at labels below 65), and extracting allows naming the data and associating it with particular measurements (e.g., if the label is 3, then we will save the data field in a measurement called Airspeed).

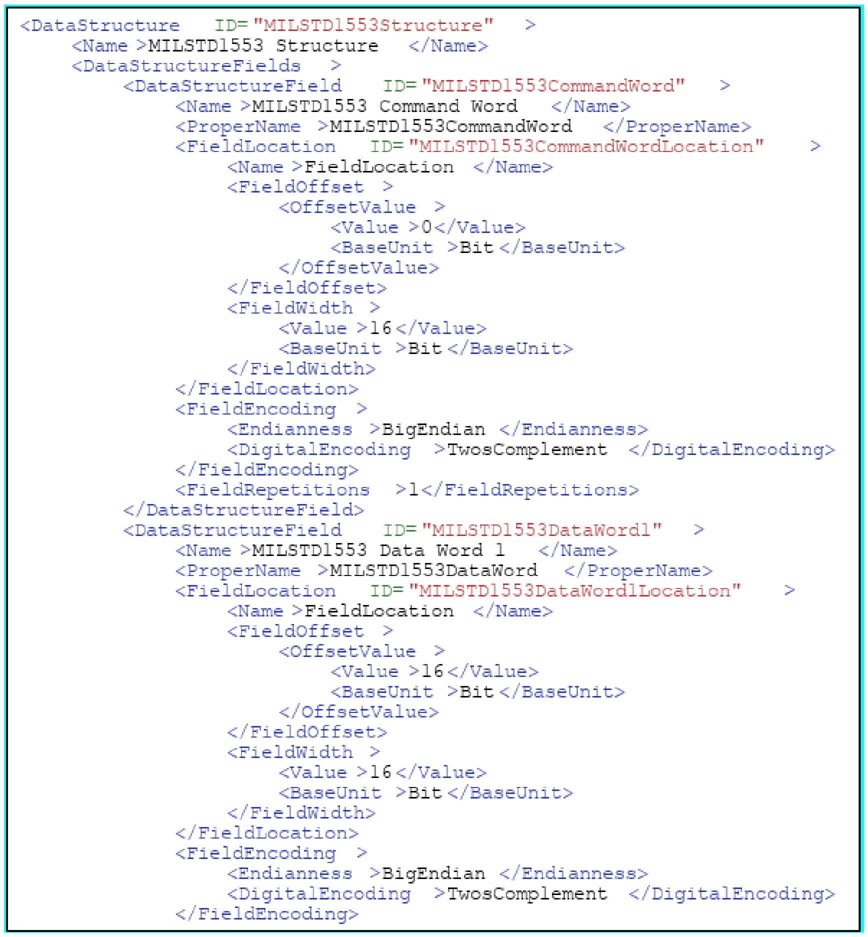
3.2.3.2 ARINC 429 Avionics Bus Capture

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

This section provides descriptions of the different ways to capture and package ARINC 429 data. Note that much of this will have been covered as part of teaching the previous sections, so this will mainly be summary.

3.2.3.3 MIL-STD-1553 Avionics Bus Capture

Moving into a more complicated bus and set of structures, next is the MIL-STD-1553 data bus. This has several structures of data, not just a 32-bit fixed structure like has been covered up to this point, and additionally, a variable number of data words that can be present. The contents of the message are determined by the command word, meaning that in order to pull out the data from the messages, some conditional decoding will need to be done. Fortunately, this is again where the MeasurementSelectors come in, as covered in Subsection 3.2.3.1.3 for the previous ARINC 429 example. The method in which the structures get used for 1553 are somewhat more complicated, however, and so we will walk through another full example as illustrated in Figure 3-17.



```
<DataStructure   ID="MILSTD1553Structure"   >
     <Name >MILSTD1553 Structure   </Name>
     <DataStructureFields   >
          <DataStructureField    ID="MILSTD1553CommandWord"    >
               <Name >MILSTD1553 Command Word   </Name>
               <ProperName >MILSTD1553CommandWord   </ProperName>
               <FieldLocation    ID="MILSTD1553CommandWordLocation"    >
                    <Name >FieldLocation  </Name>
                    <FieldOffset  >
                         <OffsetValue  >
                              <Value >0</Value>
                              <BaseUnit >Bit</BaseUnit>
                         </OffsetValue>
                    </FieldOffset>
                    <FieldWidth  >
                         <Value >16</Value>
                         <BaseUnit >Bit</BaseUnit>
                    </FieldWidth>
               </FieldLocation>
               <FieldEncoding  >
                    <Endianness  >BigEndian </Endianness>
                    <DigitalEncoding  >TwosComplement  </DigitalEncoding>
               </FieldEncoding>
               <FieldRepetitions  >1</FieldRepetitions>
          </DataStructureField>
          <DataStructureField    ID="MILSTD1553DataWord1"    >
               <Name >MILSTD1553 Data Word 1   </Name>
               <ProperName >MILSTD1553DataWord   </ProperName>
               <FieldLocation    ID="MILSTD1553DataWord1Location"    >
                    <Name >FieldLocation  </Name>
                    <FieldOffset  >
                         <OffsetValue  >
                              <Value >16</Value>
                              <BaseUnit >Bit</BaseUnit>
                         </OffsetValue>
                    </FieldOffset>
                    <FieldWidth  >
                         <Value >16</Value>
                         <BaseUnit >Bit</BaseUnit>
                    </FieldWidth>
               </FieldLocation>
               <FieldEncoding  >
                    <Endianness  >BigEndian </Endianness>
                    <DigitalEncoding  >TwosComplement  </DigitalEncoding>
               </FieldEncoding>
```

Figure 3-17. MDL Example Description of MIL-STD-1553 Data Structure

First, the DataStructures for 1553 must be defined. There are several shapes, but for learning purposes, we will use the Bus Controller to Remote Terminal (RT) transfer structure. In this case, we have a 16-bit command word, followed by 1 to 32 16-bit data words. In MDL v1.0.0, the method to describe this is to just describe the maximum length structure. Later versions of MDL support a more formal definition of what the possible shapes are, and how they are impacted by the contents of the data (e.g., if the command word is X, there are exactly 17 data words), but in this case, we'll describe the longest possible one, and just not extract any measurements from parts of the status message that would not exist.

We will start with defining the command word. We start with a new DataStructure, and add a 16-bit DataStructureField, starting at the beginning of the message, and with a ProperName of MILSTD1553CommandWord. Then, we will define all 32 of the data words. Following the command word, we add 32 more data fields, each 16 bits long, and each with a ProperName of MILSTD1553DataWord.

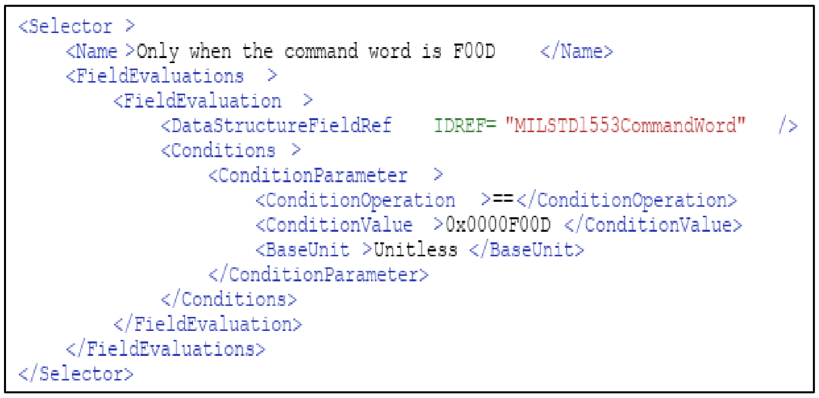
With our DataStructure in hand, we will move to describing how to extract measurements out of it. If the DAU doing acquisition was just doing a passthrough-type operation, from here on the process is the same as what was described in Subsection 3.2.3.1.1 and Subsection 3.2.3.1.2, so we will focus on how to do measurement extraction from this more complicated structure.

Like we saw before, we will make a DataStream, with a DataStreamMessage corresponding to the structure we just defined. If we were to continue later and define more of the possible shapes that will appear on the bus (RT to RT transfers, etc.), each would get its own DataStreamMessage referencing that particular structure, and the process we're about to outline would be repeated.

Within the DataStreamMessage, we'll move right into the MeasurementSelectors. Like we showed above in Subsection 3.2.3.1.4, there are practical purposes to using a filter stage separately from the extraction stage, but in this case, we will just use the MeasurementSelectors to handle what we want, which is extracting the data words based on the contents of the command word.

In order to do an illustrative example, we first need to define an entry in a notional bus catalog so that we know what we're extracting. Let's say we have a command word with the value 0x0000F00D, which is followed by six data words, where only the second and fifth are relevant for our current application. We want to describe a setup where, if the command word's value is 0x0000F00D, then we take the values from the second and fifth words and store them into measurements for further network transport. Perhaps in our case, the value 0x0000F00D is for a set of temperature sensors in an engine, and the second and fifth words correspond to the current reported values from each of those, ignoring the other data words that provide running averages or something that could be calculated later.

It's possible to notice here that our goal is an "if X, then Y" statement, and that's also how we can think about what the MDL implementation is. We'll first define a Selector stating the "if X", which in this case is "if the hex value of the command word is 0x0000F00D". As shown in Figure 3-18, there will only be one FieldEvaluation, referencing the DataStructureField of the 1553 command word, since that is the only spot that needs to be checked to know what to do with the rest of the bus message. Correspondingly, since we're just checking for equality, we have just one ConditionParameter, which is checking, in words, "Is the value of the field being looked at exactly equal to 0x0000F00D". That's all that's necessary for the Selector in this case, and the "if X" side of the condition is done.



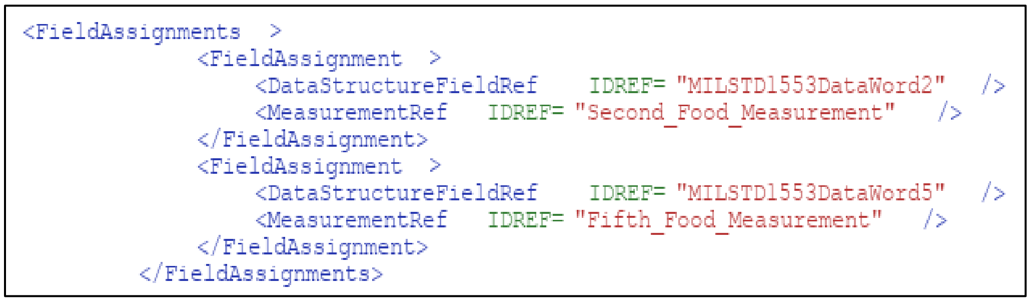
```
<Selector >
    <Name >Only when the command word is F00D    </Name>
    <FieldEvaluations  >
        <FieldEvaluation  >
            <DataStructureFieldRef    IDREF= "MILSTD1553CommandWord"   />
            <Conditions >
                <ConditionParameter  >
                    <ConditionOperation  >==</ConditionOperation>
                    <ConditionValue  >0x0000F00D </ConditionValue>
                    <BaseUnit >Unitless </BaseUnit>
                </ConditionParameter>
            </Conditions>
        </FieldEvaluation>
    </FieldEvaluations>
</Selector>
```

Figure 3-18. MDL Example Description of a MIL-STD-1553 Measurement Selector

The other piece of the puzzle is the "then Y" statement, which in this case is "then capture the second and fifth data words to use later". Our later use in this case is to slot them into a TmNS message and put them onto the network, just like we saw with the analog measurements in Subsection 3.2.2. To notate how to pull out the measurements, we need two

FieldAssignments, which each mean we're assigning the contents of a particular field to a particular measurement.

Using Figure 3-19 to illustrate for each, there are only two parts - what field we're taking the value from, and what measurement it's going into. For the second data word, we will reference the second data word of the 1553 message structure as our DataStructureFieldRef, since that's where the value is coming from, and reference the "Second_Food_Measurement" for our MeasurementRef, since that's where the data is going. Similarly, for the fifth data word, we will reference the fifth data word of the 1553 message structure, since that's where the value is coming from, and reference the "Fifth_Food_Measurement" for our MeasurementRef.



```
<FieldAssignments  >
            <FieldAssignment  >
                <DataStructureFieldRef    IDREF= "MILSTD1553DataWord2"   />
                <MeasurementRef   IDREF= "Second_Food_Measurement"   />
            </FieldAssignment>
            <FieldAssignment  >
                <DataStructureFieldRef    IDREF= "MILSTD1553DataWord5"   />
                <MeasurementRef   IDREF= "Fifth_Food_Measurement"   />
            </FieldAssignment>
        </FieldAssignments>
```

Figure 3-19. MDL Example Description of MDL-STD-1553 Measurement Field Assignments

Other contents of the bus can be handled in a similar fashion. If there are more command words with relevant data to capture, then for each command word:

- create a new MeasurementSelector;
- create a Selector within that to set up the "if this is the right command word" check;
- create one or more FieldAssignments, one for each piece of data that should be extracted.

One thing to note is that, since we defined the DataStructure with one field per data word, we are unable to reference individual bits or portions of data words, just the whole thing. Depending on the capabilities of the system, this can be handled in two ways.

First, if the DAU does support operating at this sub-data word level, we can update our DataStructure to include specific references to the portions we would like to use. The MDL structure supports having overlapping fields, so we can simply define new fields that represent subsections of the full data word, as needed. Figure 3-20 shows an example of this, defining two new 8-bit fields within a data word that could be used as an alternative shape, rather than the full 16-bit data word.

```
<DataStructureField   ID="MILSTD1553DataWord1Byte2"   >
    <Name >MILSTD1553 Data Word 1 Byte 2   </Name>
    <ProperName >None </ProperName>
    <FieldLocation   ID="MILSTD1553DataWord1Byte2Location"   >
        <Name >FieldLocation  </Name>
        <FieldOffset >
            <OffsetValue >
                <Value >24</Value>
                <BaseUnit >Bit</BaseUnit>
            </OffsetValue>
        </FieldOffset>
        <FieldWidth >
            <Value >8</Value>
            <BaseUnit >Bit</BaseUnit>
        </FieldWidth>
    </FieldLocation>
    <FieldEncoding >
        <Endianness >BigEndian </Endianness>
        <DigitalEncoding  >TwosComplement  </DigitalEncoding>
    </FieldEncoding>
    <FieldRepetitions  >1</FieldRepetitions>
</DataStructureField>
```

Figure 3-20. MDL Example Description of Sub-Data Word Level Field Assignments

The second option is to capture the full 16-bit data word, and then using DataProcessing, split apart the word into the desired components, and send those further on through the system. This type of application will be discussed in Section 3.5.

3.2.3.4 Generic Ethernet Capture

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

This section provides descriptions of the different ways to capture and package generic Ethernet data. It covers any non-fixed shape packages.

3.2.4 Negotiation

Configurations of DAUs are complicated, and device rules don't tend to leave the device, so it can be hard to arrive at a configuration file that the DAU will be happy with if doing this without asking the DAU throughout the process if it is capable of doing what the user wants it to do. This process is called negotiation (see Chapter 25 Subsection 25.4.3.3).

Outlined here is a standard workflow, where the user is using the SysMgr.

1. The user requests the device inventory from the DAU. This contains the hardware description of the DAU - its modules, submodules, connectors, ports, and other topology. This inventory description provides both the physical hardware available to be configured, and the logical ports and inputs that acquisition can be set up to use. This can also include the CaptureCapabilities, which define precisely the data shapes and formats of bus data that can be acquired.
2. The user uses the SysMgr to assign measurements and bus messages to the DAU to be acquired. These can include analog acquisitions from transducers such as thermocouples, digital acquisitions such as discretes, or data stream acquisitions such as cherry-picked avionics data or full bus captures. In some cases, the user can also define requested packaging for the DAU to use.
3. The user sends the candidate file to the DAU for validation. The DAU is then responsible for validating that what has been requested of it is something that is within its

capabilities, which includes verifying sample rates, bus capture shapes, and requested packaging (if present). Then, the DAU responds to the validation request, with one of the following.

   a. An "all good", where the user's file is directly usable.
   b. An "all good with revisions", where the DAU has made modifications to the user's file, but the resulting new file is directly usable. In this case, the SysMgr will merge in the DAU's changes to the user's current configuration.
   c. A "failure", where the DAU provides a validation report detailing the parts of the file that were not able to validate. In this case, the user will likely revise their submission and repeat until a file has been approved.

4. At the resolution of this process, whether on a single pass or a series of back and forth negotiations, the user will have a valid configuration file that can be used to configure the DAU for acquisition.

This process is similar when negotiating with a device emulator, in lieu of physical hardware being available. The inventory may not match the target hardware directly, so either the device topology that is supplied will need to be updated, or in some cases, the emulator can be programmed to match the target hardware. From that point on, the process is identical, redirecting all requests that would be pointed to physical hardware to arrive at the emulator.

3.3 Recorder

The main function of a TmNS recorder is to receive TDMs sent by DAUs over the network and to store them on its internal media. Once the TmNS messages, packages, and measurements that will be produced by the DAUs in the test are identified, they can be assigned to the TmNS recorder for recording. This is accomplished through configuration with MDL. The recorder functionality exercised is the LTC DataSink capability.

A TmNS recorder may have additional capabilities that allow for data to be extracted over the network while recording. However, the data retrieval capability is better coupled with a system that is utilizing a two-way telemetry link. Thus, this functionality will be explored in more detail in a later system capability in this handbook.

The previously mentioned example MDL file, here, contains a TmNS recorder configured for recording TDMs as an LTC DataSink.

The relevant portion of the MDL file looks like Figure 3-21.

```xml
<TmNSApp  ID="TA1_Recorder1App"  >
        <Name >TBD name </Name>
        <Description >TBD description  </Description>
        <RoleID >TA1_Rec1 </RoleID>
        <LoggingLevel  >Trace </LoggingLevel>
        <Manufacturer  >TBD Vendor Co  </Manufacturer>
        <Product >TBD App </Product>
        <ProductVersion  >1.0 </ProductVersion>
        <ConfigurationVersion   >-TBD- </ConfigurationVersion>
        <TmNSManagementResourcesVersion    >TBD version  </TmNSManagementResourcesVersion>
        <DirtyBit >false </DirtyBit>
        <TmNSRecorder  />
        <TmNSLTCDataSink  >
            <NetworkInterfaceRef    IDREF= "TA1_Recorder1IFace"   />
            <!--  TODO: subscribe to messages once defined      -->
        </TmNSLTCDataSink>
    </TmNSApp>
```

Figure 3-21. MDL Content of a TmNS Recorder Component

3.4 Network Configuration

The network switches that make up the onboard TA network infrastructure may require a specific configuration in order to support the transport of the TDMs. The most common approach for DAUs to deliver TDMs will be in the form of UDP over IP multicast. As such, the switches need to be configured appropriately in order to support the desired mode of operation for multicast packet delivery. This section will discuss the modes of multicast routing through TmNS switches and the TmNS configuration options associated with them.

In addition to multicast delivery, TA network switches play a critical role in delivering time synchronization packets across the network to ensure the sub-microsecond accuracy of connected nodes across the subnet. Network switches that are capable of operating as the GMC will have additional configuration options. These will also be discussed in this section.

The example MDL files and the associated configuration options discussed in this section are specifically related to TmNS-compliant network switches. If non-TmNS network switches are used to provide the network infrastructure for the TA instrumentation network, there are still specific requirements that they must meet. They are required to provide hardware support for the IEEE 1588-2008 protocol. Configuration of non-TmNS network switches will need to be handled out-of-band through a vendor-provided interface.

3.4.1 Multicast Behavior Through Network Switches

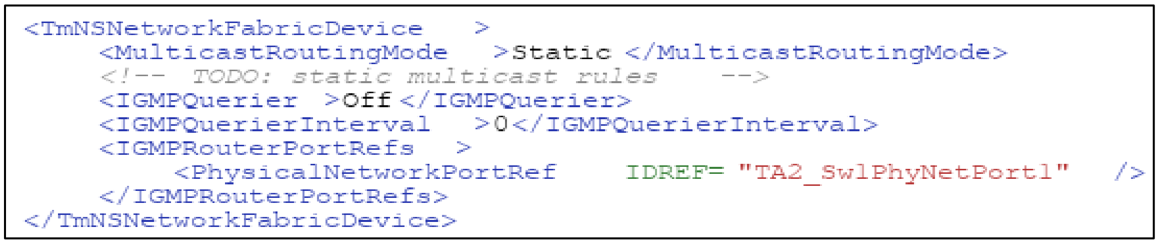
Many network switches treat multicast traffic similar to broadcast traffic in that they flood the traffic out all other ports. While this may work as a delivery mechanism, it is inefficient and can lead to throughput bottlenecks. Instead, multicast traffic delivery can be based on a static configuration of output ports per multicast group address, or it can be based on a dynamic subscription per multicast group address in the form of IGMP membership reports. A TmNS switch can be configured for either of these methods or a hybrid approach that allows both static rules and dynamic multicast subscriptions to base multicast forwarding descriptions on.

3.4.1.1 Static Multicast Routing Configuration

For static multicast configuration on switches, multicast group addresses must be explicitly defined along with each network port on the switch to which traffic for the specific group is to be delivered. These multicast routes are specified through MDL configuration. Multiple ports may be specified for a single group if necessary. Without a multicast group address being specified for a port, the switch will not forward the traffic out the port. The IGMP subscriptions from potential end nodes are ignored in a static-only approach.

The previously mentioned example MDL file, here, contains a TmNS network switch configured for static multicast routing.

The relevant portion of the MDL file looks like Figure 3-22.



```
<TmNSNetworkFabricDevice   >
    <MulticastRoutingMode   >Static </MulticastRoutingMode>
    <!--  TODO: static multicast rules    -->
    <IGMPQuerier  >Off</IGMPQuerier>
    <IGMPQuerierInterval   >0</IGMPQuerierInterval>
    <IGMPRouterPortRefs   >
        <PhysicalNetworkPortRef    IDREF= "TA2_SwlPhyNetPort1"   />
    </IGMPRouterPortRefs>
</TmNSNetworkFabricDevice>
```

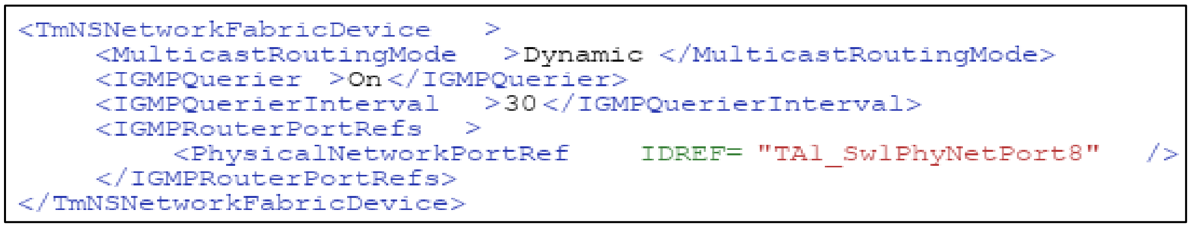
Figure 3-22. MDL Content of a TmNSNetworkFabricDevice Configured for Static Multicast Routing in the Example File

3.4.1.2 Dynamic Multicast Routing Configuration

For dynamic multicast configuration on switches, the multicast group forwarding decisions are based upon IGMP active subscribers to each multicast group. Without any subscribing nodes for a particular multicast group address, the switch will not deliver the multicast group traffic to any end node ports. Any static port configuration is disregarded. Under a dynamic multicast routing mode, the IGMP querier and query interval may need to be set. There should be at least one switch or router within a dynamic multicast routing subnet configured to be capable of performing the IGMP querier's operations. The IGMP querier can be set to "On" or to "Auto". The "Auto" setting allows the switch to participate in an election process in which one of the candidates will assume the role of the IGMP querier. For best network performance, one of the centralized, higher-speed switches should take on this role. This can be achieved by explicitly setting a specific network switch to being the querier while all others are set to "Off", or by setting a specific group of switches to "Auto" and all others set to "Off".

The previously mentioned example MDL file, here, contains a TmNS network switch configured for dynamic multicast routing.

The relevant portion of the MDL file looks like Figure 3-23.



```
<TmNSNetworkFabricDevice   >
    <MulticastRoutingMode   >Dynamic </MulticastRoutingMode>
    <IGMPQuerier  >On</IGMPQuerier>
    <IGMPQuerierInterval   >30</IGMPQuerierInterval>
    <IGMPRouterPortRefs   >
        <PhysicalNetworkPortRef    IDREF= "TA1_SwlPhyNetPort8"   />
    </IGMPRouterPortRefs>
</TmNSNetworkFabricDevice>
```

Figure 3-23. MDL Content of a TmNSNetworkFabricDevice Configured for Dynamic Multicast Routing in the Example File

3.4.1.3 Static and Dynamic Hybrid Multicast Routing Configuration

The hybrid approach provides a combination of both static and dynamic methods. In this mode, either method can yield a multicast forwarding decision. The switch will forward traffic for any static multicast groups assigned. Static group assignments are provided through the MDL configuration, and any dynamic group assignments learned through IGMP will also be forwarded to the subscribing port. The MDL configuration appears similar to that of the static multicast routing configuration with the lone exception being the <MulticastRoutingMode> value being set to "StaticAndDynamic".

The previously mentioned example MDL file, here, contains a TmNS network switch configured for the hybrid static and dynamic multicast routing.

3.4.2 Time Synchronization and Configuration Parameters for Network Switches

The TA's network switches need to support the IEEE 1588-2008 precision time protocol (PTP) in some form, either as a boundary clock or a transparent clock. Hardware support for the protocol at each network interface (e.g., network port) is required in order to achieve the highest time synchronization accuracies across the network. Because the DAUs are distributed throughout the network and timestamp the data as it is acquired, it is crucial that the DAUs maintain a high degree of time synchronization across the network. Having hardware support across the entire network transport path provided by the network switches can allow for sub-microsecond clock synchronization accuracies across all IEEE 1588 slave devices in the network.

Some of the network switches will also have the ability to operate as the IEEE 1588-2008 GMC in the network. For GMC operation, the network switch will need to have hardware support for a GPS signal input by which it can synchronize its local clock to GPS. Only one GMC is required in the TA network. It can be either a function of a network switch component, or it can be a stand-alone GMC device.

The previously mentioned example MDL file, here, contains a TmNS network switch configured to operate as the IEEE 1588 GMC.

3.5 Data Processing Applications

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

Data processing of the acquired data can be accomplished either live during flight test mission or during post-flight data analysis. Because data is being acquired and recorded, it is expected that there will also be an application to process the data from the recorder. Data processing applications will be required to process MDL in order to decode the data from within the TDMs. This section will discuss the basic types of data processing algorithms and provide examples for how they are described in MDL.

For live data processing, a two-way telemetry capability is desired in order to allow for requesting of data over the two-way link. This capability is discussed in a later section.

3.6 System Manager

For the data acquisition capability, the SysMgr application provides a GUI for the user to be able to generate the MDL configuration files associated with the components needed for achieving the data acquisition capability. This includes describing measurements and defining the message and package structures of the TDM to be used in the test.

Without a two-way telemetry capability, a SysMgr application running on a ground-based platform will not be able to communicate with the components on the TA network during flight. However, the SysMgr can be used to configure, monitor, and verify TmNS component functionality in the lab or in a preflight checkout scenario.

CHAPTER 4

Two-Way Telemetry Capability

A TmNS system with a two-way telemetry capability consists of two or more TmNS radio components for the wireless communications of the two-way link. Optionally, an LM component may be utilized within the system in order to provide a dynamically allocated transmission schedule for each of the TmNS radio components utilizing the two-way telemetry channel.

This chapter will cover the RF networking concepts that are specific to the TmNS. It is important to have a working knowledge of these concepts in order to understand the RF network configuration options available and the system performance impacts associated with those configuration options. It will also cover both static and dynamic schedule allocations of the RF network as well as radio handoff scenarios.

Figure 4-1 highlights the RAN-related components of a TmNS system with a two-way telemetry capability. The details of this diagram will be expanded upon in the following handbook subsections associated with this capability.

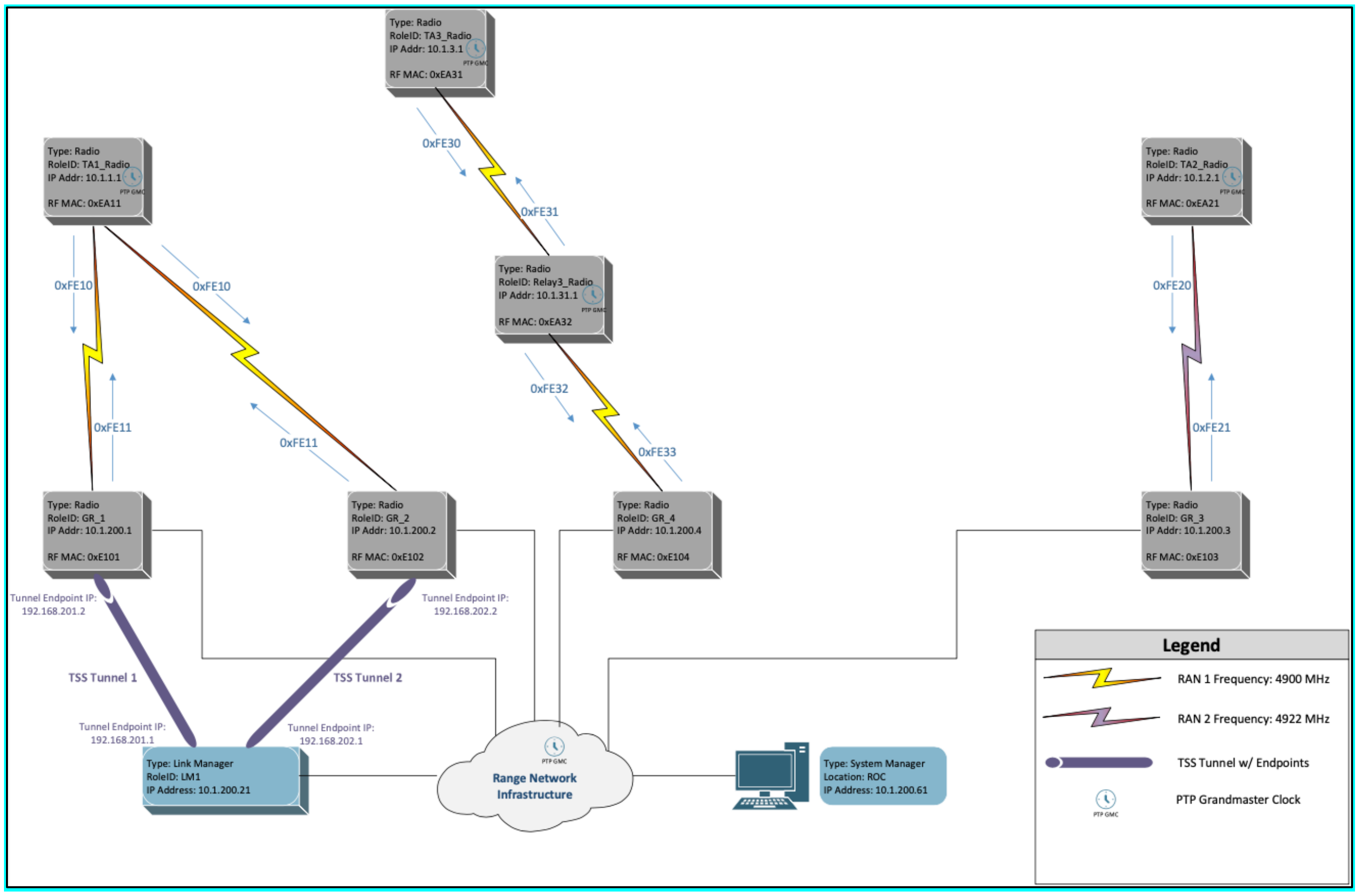


Figure 4-1. RAN Components

4.1 RF Network Concepts

The TmNS defines a two-way, time division multiple access (TDMA) telemetry channel in Chapters 27 and 28. A channel consists of a single frequency by which TmNS radios both transmit to and receive from. Because of its TDMA structure, multiple TmNS radios may transmit over the same RF channel, but only one radio may transmit at a time.

A RAN consists of an RF channel and the components and configuration parameters associated with the RF channel. Over a single RAN (RF channel), multiple independent tests can be conducted simultaneously if the RF channel is properly shared amongst the participating TmNS radios. As long as there is only a single transmitting radio on the RF channel at any one point in time, the RF channel can be shared by multiple test programs. Sharing of RF channels can be accomplished either with well-known static transmission schedule assignments or with dynamically assigned transmission schedules.

This section is designed to provide the reader with enough understanding of the concepts and capabilities of the RF network in order to be able to design and configure a RAN. The following concepts will be covered.

- RAN Configuration Parameters
- RF Multicast
- Handoffs

Where applicable, MDL examples will be provided to aid in understanding of configuration parameters.

4.1.1 RAN Configuration Parameters

The MDL configuration files provide RAN configuration for those components that make up and utilize the RAN, namely the TmNS radio and TmNS LM components. Key RAN configuration parameters are described in more detail in the following subsections.

The previously mentioned example MDL file, here, contains a RAN configuration.

The relevant portion of the MDL file is provided in Figure 4-2.

```
<RANConfiguration   ID="RANConfig1" >
        <Name >RANConfig1 </Name>
        <Description >RAN configuration parameters for RAN1    </Description>
        <LinkAgentConnectionTimeout    >30</LinkAgentConnectionTimeout>
        <LinkAgentConnectionEncryptionEnabled    >true </LinkAgentConnectionEncryptionEnabled>
        <TSSTunnelEncryptionEnabled    >true </TSSTunnelEncryptionEnabled>
        <CenterFrequencyHz  >4900000000 </CenterFrequencyHz>
        <ModulationType  >SOQPSK-TG </ModulationType>
        <EpochSize >100</EpochSize>
        <LDPCBlocksPerBurst  >4</LDPCBlocksPerBurst>
        <MaxGuardTimeSec  >0.001 </MaxGuardTimeSec>
        <RadioControlLoopDSCPRef    IDREF= "Diffserv_DSCP56_NetworkControl"    />
        <RANCommandControlDSCPRef    IDREF= "Diffserv_DSCP48_InterNetworkControl"    />
        <RadioGroups >
            <RadioGroup  ID="GROUND_GROUP_1"  >
                <Name >RadioGroup_Ground  </Name>
                <Description >The members of this radio group are ground radios for Test 1      </Description>
                <GroupRFMACAddress  >65040 </GroupRFMACAddress>
            </RadioGroup>
```

Figure 4-2. RAN Configuration Parameters

4.1.1.1 Frequency

The RAN's operational frequency is that by which all TmNS radios within the RAN are tuned to transmit and receive on. If an LM is present within the RAN, it allocates dynamic transmission schedules for this particular frequency only. When a radio is configured, it begins listening for transmissions on this frequency. Chapter 27[9] recommends RAN frequencies to be either 4,900.0 MHz or 4,922.0 MHz.

The corresponding MDL element name is <CenterFrequencyHz>.

4.1.1.2 Epochs

The RAN epoch provides the structure by which transmission opportunities can be assigned to various radios within the RAN. Each RAN operates with a specific epoch size. Transmission opportunities maintain start and stop times relative to their position within the epoch structure. The next epoch begins immediately when the former epoch ends.

Time synchronization is critical across all TmNS radios in order to prevent multiple TmNS radios from transmitting at the same time on the same RAN. This is achieved through the radios either locking directly to GPS or to an IEEE 1588 GMC that is locked to GPS. The start of every second always represents the start of a new epoch. There are different possible sizes of epochs that may be used. The TmNS limits the valid values of the epoch size to the following values: 10 ms, 20 ms, 25 ms, 40 ms, 50 ms, 100 ms, 125 ms, 250 ms, 500 ms, and 1,000 ms. Regardless of which epoch size is chosen, all values are factors of 1,000 ms, which allows for the start of the first instance of the epoch within a second to always begin at the start of the second (e.g., at the second roll-over event).

The default epoch size is 100 ms. There are trade-offs for increasing or decreasing the epoch size. Increasing the size requires additional transmission opportunity assignments to be made, but this allows for a finer tuning of bandwidth allocation within the epoch. Larger epoch sizes will take longer to adjust the overall schedule in response to dynamically changing bandwidth requirements. If guard times are imposed between transmission opportunities for different radios, then smaller epoch sizes may see an increase in wasted bandwidth due to maintaining guard times within the allocated schedule. In general, larger transmission opportunities allow for a more efficient use of the available bandwidth, assuming other latency constraints across the system can still be met.

Transmission opportunity assignments within an epoch structure are lease-based and have a lease time that specifies the number of consecutive epochs that the transmission opportunity is valid before it expires, up to 254 repetitions. The lease time is referred to as the "TxOp Timeout" in the TmNS. A TxOp Timeout value set to 255 is a special case that represents an infinite, never-expiring transmission opportunity assignment. This is used when configuring TmNS radios to utilize a static transmission schedule where a radio may contain from one to infinite transmission opportunities. An LM can recall any number of transmission opportunities via a schedule change. An LM may provide some transmission opportunity assignments that are infinite as well as other ones that are temporary as part of its dynamic scheduling algorithm.

[9] Range Commanders Council. "Radio Frequency Network Access Layer." In *Telemetry Standards*. RCC 106-20 Chapter 26. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

The corresponding MDL element name is <EpochSize>, and the units are in milliseconds.

4.1.1.3 LDPC Blocks Per Burst

The RF burst format contains a preamble, and attached synchronization marker, and then a number of low-density parity check (LDPC) codeblocks. The number of LDPC codeblocks contained within the burst is configurable, from 1 up to 16. When a TmNS radio has data to send, the data is packed into LDPC codeblocks, and then during the radio's allocated transmission opportunity, it generates an RF burst to transmit the data in the LDPC codeblocks. For each burst, the configured number of LDPC codeblocks will be transmitted regardless of data content. A larger number of LDPC codeblocks per burst may reduce the overall number of bursts needed, which improves efficiency due to the reduction of overhead bits transmitted with each burst. However, as the number of LDPC codeblocks increases, so too increases the potential for codeblocks to be sent that do not contain any data, thus decreasing the overall efficiency of the RAN.

Each LDPC codeblock is 512 bytes in length. The max data payload size that can be carried in a codeblock is 494 bytes in an unprotected RF media access control (MAC) frame or 478 bytes in an AES-CCMP encrypted RF MAC frame.

The number of LDPC codeblocks per burst also affects the minimum size of a transmission opportunity assignment. For every LDPC codeblock within the RF burst, an additional 512 bytes are transmitted. This has some impact on the scheduling granularity.

The corresponding MDL element name is <LDPCBlocksPerBurst>.

4.1.1.4 Guard Times

Guard times are purposeful gaps in the transmission schedule of the RAN's epoch structure to ensure that transmissions from a radio on one side of the test range don't collide with a later transmission from a radio on the other side of the test range. This value is likely to be a range-dependent setting based on the size of the range.

A guard time is required to be observed between transmissions from different radios within the RAN. For each transmission window within the epoch, assume a guard time between each window. Smaller, more frequent transmission opportunity assignments for radios in the RAN will encounter a larger number of guard times applied across the epoch. Each guard time applied represents some portion of available bandwidth that is being forfeited. The LM scheduling algorithm provides the guard time spacing between transmission opportunity allocations that it provides to its different radios.

The guard time setting can quickly reduce the bandwidth efficiency of the RAN. As a simple example, assume a guard time setting of 1 ms. For a pair of radios that communicate with each other over two transmission opportunities each during a 100 ms epoch, there would be four distinct transmission windows, two per radio. With the 1 ms guard time between radio transmissions, this scenario would represent a 4% loss in efficiency due to guard time enforcement. Making room in the schedule for two more radios that also receive two transmission opportunity assignments a piece, the guard times would increase to an 8% loss in efficiency. Thus, the smallest reasonable value of guard time is best. If guard times are not required, this value may be set to zero.

The corresponding MDL element name is <MaxGuardTimeSec>, and the units are in seconds.

4.1.1.5 RF Network Message Priorities on the Network

Chapter 24 defines RF Network Messages (RFNMs), and they are used for communication between LMs and TmNS radios. The LM provides dynamic transmission schedule updates via RFNMs. The radios send link information such as queue status levels as well as receiver statistics to the LM via RFNMs. The LM uses the information from the radios to determine if any changes should be made to the current RAN transmission schedule based on current radio conditions. The LM also uses the receiver statistics reported by the radios to perform automatic handoffs if authorized to do so.

The RFNMs are crucial to establishing and maintaining the RF link connectivity; therefore, they should be given an elevated DiffServ Code Point (DSCP) priority to prevent them from being queued behind lower-priority traffic. This is particularly critical when sending dynamic schedule updates to an airborne radio, as the path must first traverse the RF link via a radio on the ground network. Failure to deliver RFNMs in a timely manner will affect the overall responsiveness and effectiveness of the RAN. Because of this, the following two MDL elements should reference the DSCP table entry with the highest DSCP value:

- <RadioControlLoopDSCPRef>
- <RANCommandControlDSCPRef>

4.1.1.6 Radio Groups

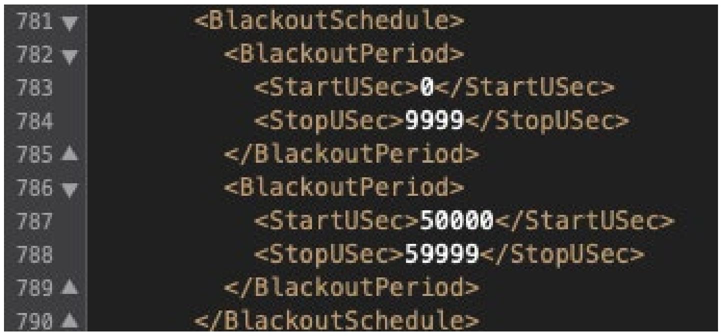
Radio groups are defined as part of a RAN's configuration. These group definitions provide an RF MAC address that can be referenced by a link definition. These will be discussed in further detail in Subsection 4.1.2.

4.1.1.7 Blackout Schedules

For purposes of dynamic scheduling, the entire epoch is available for an LM to allocate unless explicitly configured to not schedule a particular portion of the defined epoch structure. This is done through providing a blackout schedule inside the RAN. Within the schedule are defined blackout periods with a start and stop time relative to the epoch structure in which the LM will not allocate transmission opportunities.

For most implementations, a RAN will likely be entirely defined with static schedules for the entire epoch, or it will be entirely dynamically allocated by an LM. There may be certain cases in which the RAN may contain both. Keep in mind that for every blackout period defined, it is lost bandwidth within the RAN unless an active radio is statically configured to utilize that period.

The previously mentioned example MDL file, here, contains a RAN configuration with blackout schedules The MDL snippet in Figure 4-3 shows the relevant portion of the MDL example file.



```
        <BlackoutSchedule>
          <BlackoutPeriod>
            <StartUSec>0</StartUSec>
            <StopUSec>9999</StopUSec>
          </BlackoutPeriod>
          <BlackoutPeriod>
            <StartUSec>50000</StartUSec>
            <StopUSec>59999</StopUSec>
          </BlackoutPeriod>
        </BlackoutSchedule>
```

Figure 4-3. MDL Content of a RAN's Blackout Schedule for Dynamic Scheduling in the Example File

4.1.2 RF Multicast

The concept of RF Multicast was developed for the TmNS. It is analogous to IP multicast in the sense that a transmitting entity can transmit to zero or more receiving entities without knowledge of the number of receiving entities available.

This section will describe RF MAC addresses, including RF Group MAC addresses, RF links, the point-to-multipoint and point-to-point nature of links, and TSS tunnels. Understanding RF Multicast is crucial to understanding the details and mechanics of the different radio handoff scenarios; thus, it precedes the Handoffs section.

4.1.2.1 RF MAC Address

An RF MAC address is a 16-bit identifier that is assigned to the RF network interface of a TmNS radio. It performs a similar role as Ethernet MAC addresses do in typical wired Ethernet networks. The RF MAC addresses are used to define the endpoints of a wireless transmission over the RF network, from source to destination.

Table 28-1 of Chapter 28[10] provides the recommended guidance for the values of RF MAC addresses. A portion of the RF MAC address range is allocated for RF Multicast group addresses. The 16-bit address space has been divided into a 4-bit vendor ID field followed by a 12-bit RF interface field. The recommended vendor ID field values are provided in Table 4-1.

Table 4-1. RF MAC Header Vendor IDs

Vendor IDs	Description
4'b 0000	Reserved
4'b 0001 – 4'b 1101	Vendors
4'b 1110	Experimental
4'b 1111	Multicast

The TmNS has not allocated any specific vendor a specific vendor ID value. This may be done in a later version of the TmNS, or it can be handled programmatically by a range. A radio's RF MAC address is assigned to it during configuration through the MDL configuration file.

[10] Range Commanders Council. "Radio Frequency Network Management." In *Telemetry Standards*. RCC 106-20 Chapter 28. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

4.1.2.2 RF Links

According to the TmNS, an RF link is comprised of a source and destination pair of RF MAC addresses. By this definition, a single RF link only provides transmission in one direction. In order to establish two-way wireless communication over the RF network, a second RF link is needed. Two links can be seen between the radio components *TA2_Radio* and *GR_3* in the layout shown in Figure 4-4.

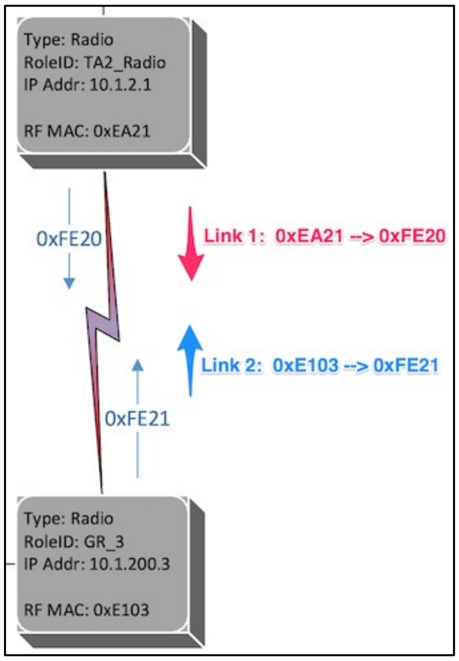


Figure 4-4. Two RF Links Required for Two-Way Communications

Care must be taken when describing a disruption in communications because two-way communication requires two distinct RF links. The term "link" is overloaded and used to mean different things. For instance, during test operations, operators may say that "the link is up" or "the link dropped out." Here, the term "link" is being used to describe the overall two-way communications capability. The implication here is that when "the link is up", both of the two RF links are functioning properly for transmitting and receiving to and from the two radios. When "the link is down," the reason it is down is not immediately known as it could be due to an issue with only one of the RF links.

When a radio is transmitting, it does so in RF bursts, where each burst contains a configured number of LDPC code blocks. Each LDPC code block transmitted begins with an RF MAC Header as defined in Chapter 27, which is shown in Figure 4-5.

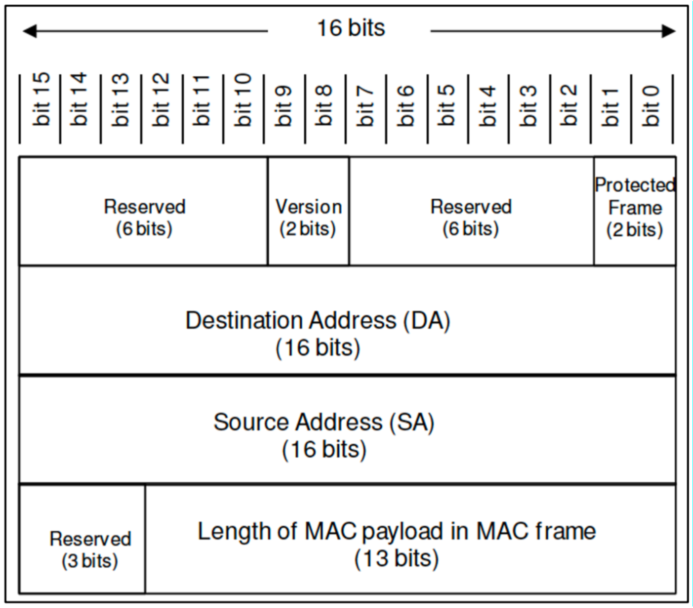


Figure 4-5. RF MAC Header Structure

The RF MAC Header contains both the destination and source RF MAC addresses. The source RF MAC address identifies which TmNS radio transmitted the RF burst. The destination RF MAC address identifies the radio or group of radios that the LDPC code block is destined for. When a radio begins receiving a wireless transmission, it inspects the RF MAC Header of each LDPC code block to see if the code block is destined for itself, either through being addressed to its own RF MAC address or through being addressed to a group RF MAC address that it is subscribed to. If so, it receives and processes the payload data of the LDPC code block. If not, then the receiving radio ignores the rest of the LDPC code block.

The destination RF MAC address determines whether or not the communication is point-to-multipoint or point-to-point. These two are differentiated in the subsections below.

4.1.2.2.1 Point-to-Multipoint Links

Point-to-point RF links capitalize on the fact that the wireless network is one common transmission medium shared across the RAN. After all, the RF transmissions are broadcasted over the RAN. Because of this, care is taken to schedule the transmission opportunities per radio in such a way so as to not overlap any transmission opportunities across multiple radios within the same RAN. The multipoint nature of these links is analogous to IP multicast in that multiple radios can be configured to receive the same RF transmissions. The transmitting radio is not aware of how many radios are receiving its transmissions, if any at all.

There are specific scenarios where it is beneficial to have multiple receiving radios for a single link. This is most often required for the downlink communication link where multiple ground-based TmNS radios may be used to track a single airborne TmNS radio across the test range. Having multiple ground-based radios receiving on the same destination group RF MAC address allows for the various radio handoff-type scenarios, discussed in a later section.

As for uplink communication links from a ground-based TmNS radio to a single airborne radio, it could be configured as a point-to-point link. However, there is nothing that would prevent a point-to-multipoint link to be utilized with the airborne radio being the only radio within the receiver group for the link. From a link group management standpoint, it may be a simpler approach to make all links, regardless of direction or anticipated number of recipients, multipoint links that use group RF MAC addresses for link destinations. Utilizing group RF MAC addresses, it becomes very easy to add a new radio to a group in order to start receiving transmission from an existing radio. Because of this, utilizing point-to-multipoint links is the recommended approach for link assignments.

The example system figures and files utilize point-to-multipoint link, even when there is only a single recipient of the link.

4.1.2.2.2 Point-to-Point Links

Point-to-point RF links are intended to be received by one specific TmNS radio. The destination RF MAC address used in the link definition is specific to a particular radio. Specifying point-to-point links is valid; however, it should only be utilized if there is good reason to not utilize a point-to-multipoint link assignment.

4.1.2.2.3 Best Practices for Defining RF Links

When defining RF links for two-way telemetry, point-to-multipoint links provide the greatest flexibility for system capability. They support multiple radios receiving the same transmission, such as will be the case of a TA radio’s transmission to several ground station radios, which is the basis for both a packet-based best source selector and radio handoff capabilities. The same concepts apply to radio transmissions regardless of the number of receiving radios, be it zero, one, or more than one. For these reasons, it is recommended to always utilize point-to-multipoint links unless a specific case requires something different.

To properly create the needed RF groups, one must consider only the destination RF MAC address of the link(s). First, identify all radio assets that will be associated with the two-way links. Then, separate the radios into groups based on the desired communication direction. This often will result in two groups: one group containing only the TA radio, and one group containing all of the ground-based radio assets. In order for the two groups to communicate, a path must be defined from each group destined to the other group. Each of these paths is associated with a unique destination group RF MAC address. Figure 4-6 highlights this process.

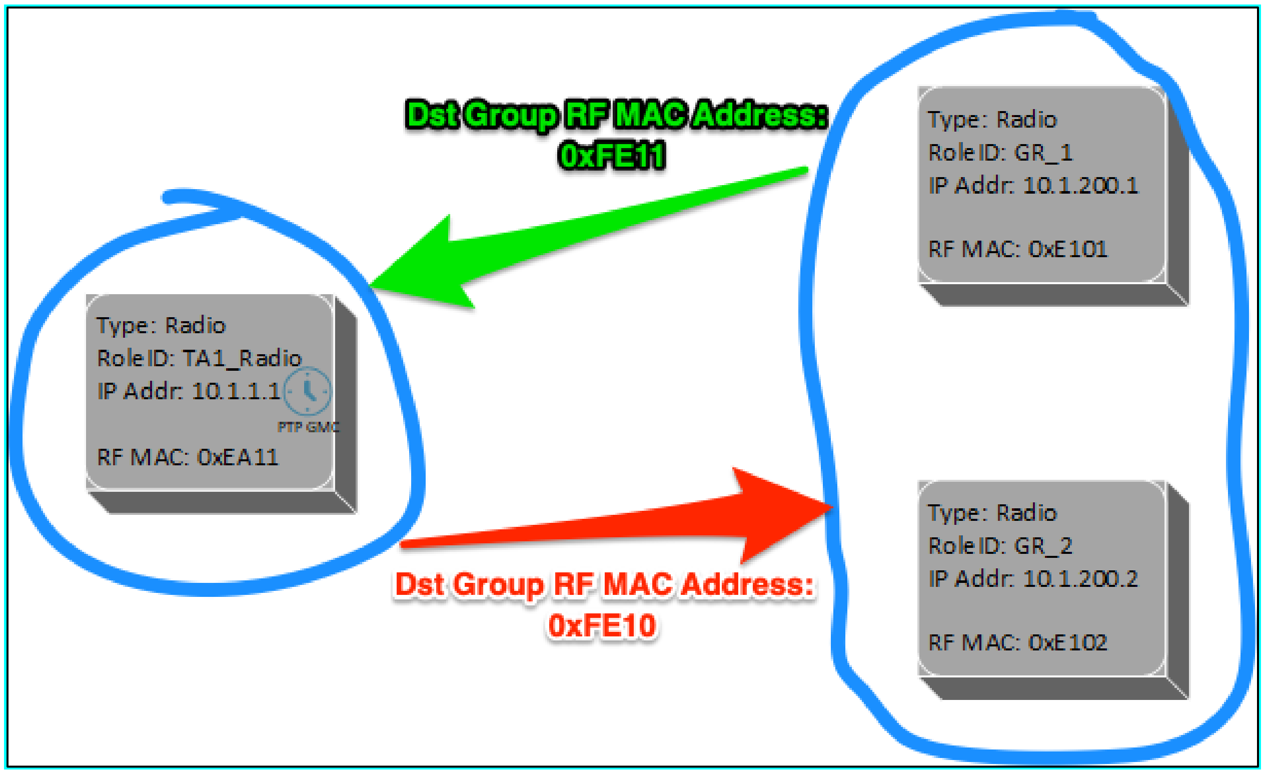


Figure 4-6. Creating RF Links for Two-Way Telemetry

To define the needed transmission links, a unique link will be created for each radio in each group, utilizing the RF MAC address of each radio as the source RF MAC address of the link, and the link destination being the group RF MAC address that is transmitted to by all radios in the radio collection. The transmission links can be seen in Figure 4-6 with the red and green arrows. From the example, TA1_Radio can transmit to the group RF MAC address of 0xFE10, and both GR_1 and GR_2 are able to transmit to the group RF MAC address of 0xFE11.

In order to receive an RF transmission, all radios in the collection will need to subscribe to the destination group RF MAC address being sent to the collection. This is done within each radio's configuration under the <JoinRadioGroupRefs> element. From the example, TA1_Radio must join (e.g., subscribe to) the radio group reference to the group RF MAC address 0xFE11, and similarly, GR_1 and GR_2 must each join the radio group reference to the group RF MAC address of 0xFE10.

4.1.2.3 TmNS Source Selector Tunnels

The TSS tunnels facilitate the use of RF Multicast across the test range. A TSS tunnel is used to encapsulate IP network traffic to or from an airborne network as it is routed across the ground infrastructure network of the range. The TSS tunnels may be established between an LM and a TmNS radio or between two LMs. Each tunnel endpoint provides a virtual network interface on the device that is used for routing traffic through the tunnel. A single TSS tunnel consists of two network endpoints that are connected via a TCP connection. The virtual network interface's IP address is known by the other side of the tunnel and is used as a next hop route for traffic needing to pass through the tunnel. Thus, the virtual subnet of the tunnel is a unique subnet in the TmNS network. Each TSS tunnel that a single LM establishes will be a unique IP subnet that can be used for routing network traffic through.

The TSS Data Messages are passed between the tunnel endpoints. Each TSS Data Message consists of a small message header and an encapsulated Ethernet frame. The encapsulated Ethernet frame is the message to be routed to or from the airborne network. The TSS Data Message structure as defined in Chapter 24 is pictured in Figure 4-7.

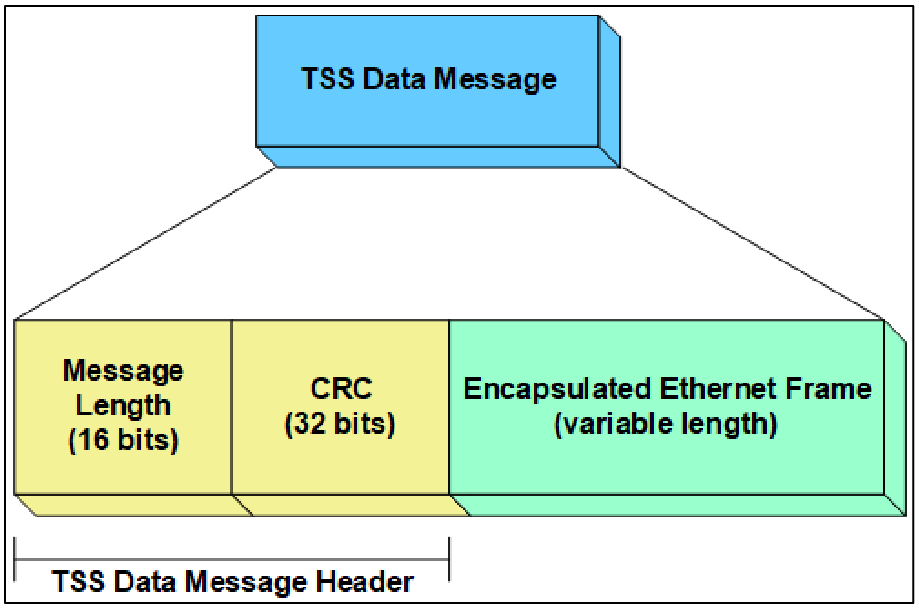


Figure 4-7. TSS Data Message Structure

When a network packet is to be forwarded through the TSS tunnel, the entire Ethernet frame is encapsulated into the TSS Data Message. An associated frame ID is generated based on a cyclic redundancy check (CRC) calculation performed on the encapsulated Ethernet frame, beginning with the IP header of the frame, and this CRC value is placed in the TSS Data Message header, along with a message length field. When a TSS Data Message is received by a tunnel endpoint, the encapsulated Ethernet frame can be extracted and routed to its next hop destination as needed. Tunnel endpoints can use the CRC-based frame ID to check for other instances of the encapsulated Ethernet frame that may have already been received.

The presence of TSS tunnels in the system allows for the use of RF multicast, which by design, will place multiple copies of the received data onto the ground infrastructure network. At an LM where multiple TSS tunnel endpoints converge, the LM is able to remove any duplicated packets received such that only a single instance of the received packet is forwarded to the destination. The tunnels also enable the ability for seamless, automatic A2A handoffs.

4.1.3 Handoffs

Handoffs refer to the ability to change the actively scheduled TmNS radio pairs that are communicating with one another during flight operations. Through the use of group RF MAC addresses, some handoffs can occur without a loss in data. There have been three types of handoff events identified. Each handoff type (A2A, N2N, and R2R) is similar in concept but different in purpose and system impacts.

Handoff capabilities are best realized in conjunction with the use of an LM performing dynamic schedule allocations to the different radios. As such, see Subsection 4.3.2 for specific LM details.

Each handoff type is described in more detail in their respective subsections below.

4.1.3.1 A2A Handoff

An A2A handoff consists of a TA-based TmNS radio and multiple ground-based TmNS radios operating within a single RF network. By leveraging RF multicast and group RF MAC addresses for the RF transmission endpoints, multiple ground-based TmNS radios can be configured to receive the transmissions from the TA-based TmNS radio. This allows for greater coverage across the test range. Though multiple radios may be able to simultaneously receive the transmissions from the TA's radio, only one transmission path exists from the collection of ground-based radios to the TA radio at any one given time.

An example of the A2A handoff can be seen in Figure 4-8 and Figure 4-9. The figures depict a TA with an omnidirectional antenna for transmitting to the ground. This pattern allows for multiple ground-based radios to receive the same transmission. While both ground-based radios can receive the transmission, only the first has been allocated a transmission schedule by the LM.

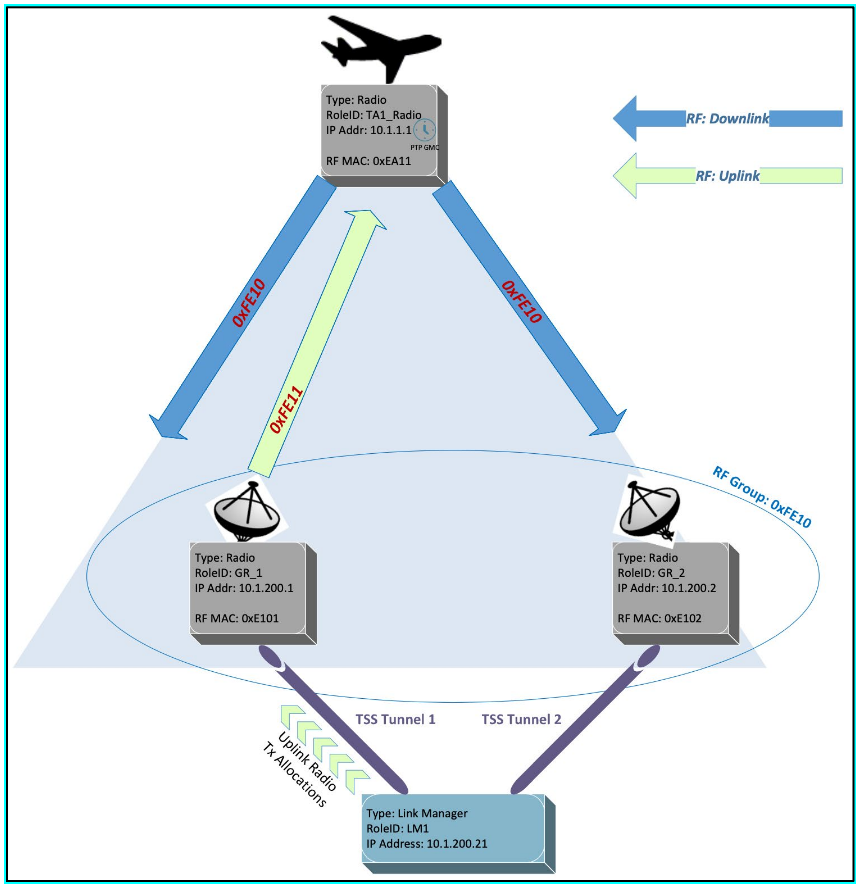


Figure 4-8. A2A Handoff: Pre-Handoff State

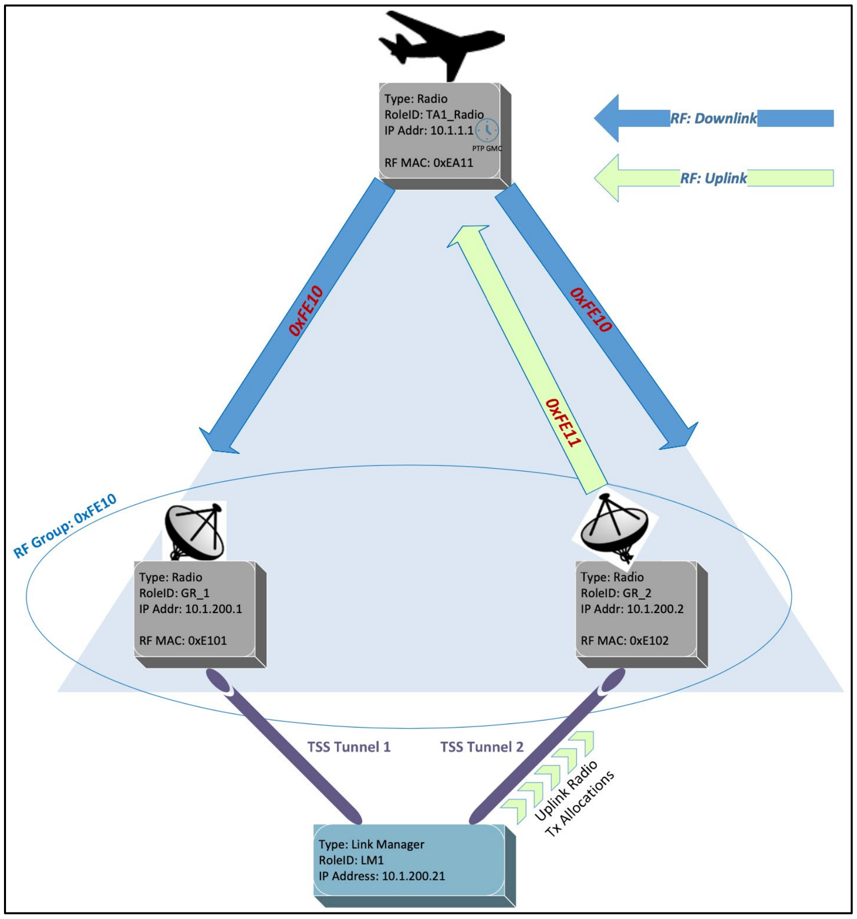


Figure 4-9. A2A Handoff: Post-Handoff State

As the TA moves across the range, an A2A handoff may be pertinent in order to maintain the two-way link. In the example, an A2A handoff is performed in order to change the uplink path to begin using the new ground-based radio, as is displayed in Figure 4-9. Due to the use of multicast RF, there is no change to the TA-based radio during the handoff process.

The management of the selected uplink is maintained by the LM for the RF network. As it dynamically allocates schedules for radio transmissions, it ensures that only a single radio within the group that share the common group RF address (both for receiving and transmitting) will have an active transmission schedule at a time.

The LM plays a crucial role in the overall handoff process. It establishes a TSS tunnel connection with each TmNS ground radio. The TSS tunnel is an IP tunnel that is used when routing data from the airborne TA network to the ground network or vice versa. For network traffic from the TA network, it is expected that at times the network traffic may be received at multiple ground-based radios. Data from these radios is sent through the TSS tunnel to the LM, at which time the LM receives the traffic and determines whether or not to continue routing the packet or not. If a packet is determined to be a duplicate of an already received and routed packet, then the LM will simply discard the duplicate, thus removing any duplicated traffic from the network.

The LM also serves as an IP router with a route to the TA network. The actual route from the LM to the TA network is dependent upon which ground-based radio is currently selected as the active uplink radio, something that the LM is aware of. The LM routes the network traffic destined for the TA network over the TSS tunnel to the ground-based radio that it is currently supplying the transmission allocation schedule to.

During the handoff process, the LM changes the route for TA network traffic to go through a different TSS tunnel. Once traffic is diverted, the LM removes the transmission schedule from the first ground-based radio and then gives one to the new ground-based radio. At this point the handoff is complete.

The A2A handoff process is carried out without any direct interaction with the TA-based radio. There is no configuration change needed to the TA-based radio, which allows for a seamless, lossless handoff scenario.

4.1.3.2 N2N Handoff

An N2N handoff also consists of a TA-based TmNS radio and multiple ground-based TmNS radios operating, but unlike the A2A handoff, this handoff event includes a frequency change by which the TA-based TmNS radio communicates with the ground-based TmNS radios. Two LMs are required for this type of handoff, each one managing a different RF network.

An N2N handoff is useful for transferring the RF network bandwidth requirements associated with one test mission to a different RF network. This may be done in order to increase overall throughput performance of the test mission by handing off to a less crowded RF network, or it may be done to provide enough bandwidth resources for a new test mission with large bandwidth requirements greater than what was available prior to moving some test missions from one RF network to another.

A user of the SysMgr manually initiates N2N handoffs. Care should be taken when initiating the N2N handoff in order to prevent a loss in communication with the TA-based radio. The handoff should be initiated whenever the ground-based TmNS radio from the initial RF network (e.g., frequency) and the ground-based TmNS radio from the next RF network are both within range of the TA-based TmNS radio. The TA-based TmNS radio will receive a new transmission schedule using the new RF network frequency that replaces all of its former transmission schedules associated with the old RF network frequency. Once these schedule changes are complete, the TA-based TmNS radio will be tuned to receive transmissions on the new RF network frequency. While this change is occurring on the TA-based TmNS radio, the second LM is attempting to connect to the TA radio through the post-handoff ground-based radio. Once the TA radio has successfully changed frequencies, the two-way link on the new RF network frequency can then be established.

An example of the N2N handoff can be seen in Figure 4-10 and Figure 4-11. Prior to the N2N handoff, the TA1_Radio transmits on the RAN 1 frequency via its omnidirectional antenna, which is depicted in Figure 4-10. Two-way connectivity is established between TA1_Radio and GR_1. Though GR_3 is in range of TA1_Radio's transmissions, it does not receive those transmissions because it is configured for the RAN 2 frequency.

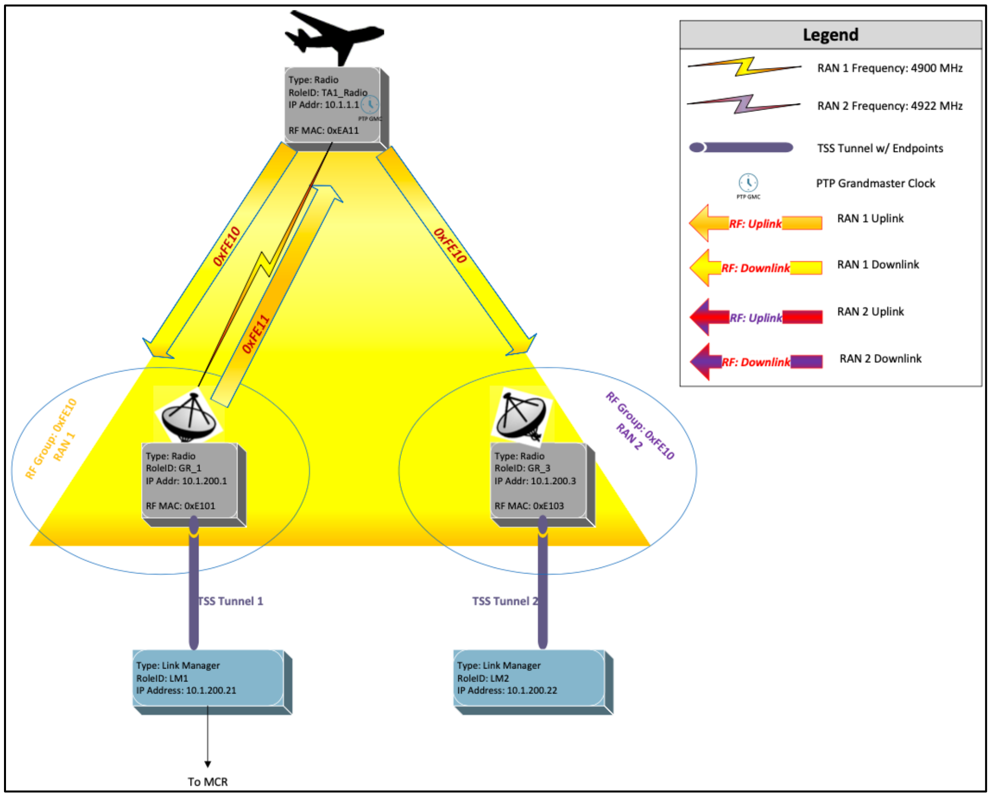


Figure 4-10. N2N Handoff: Pre-Handoff State

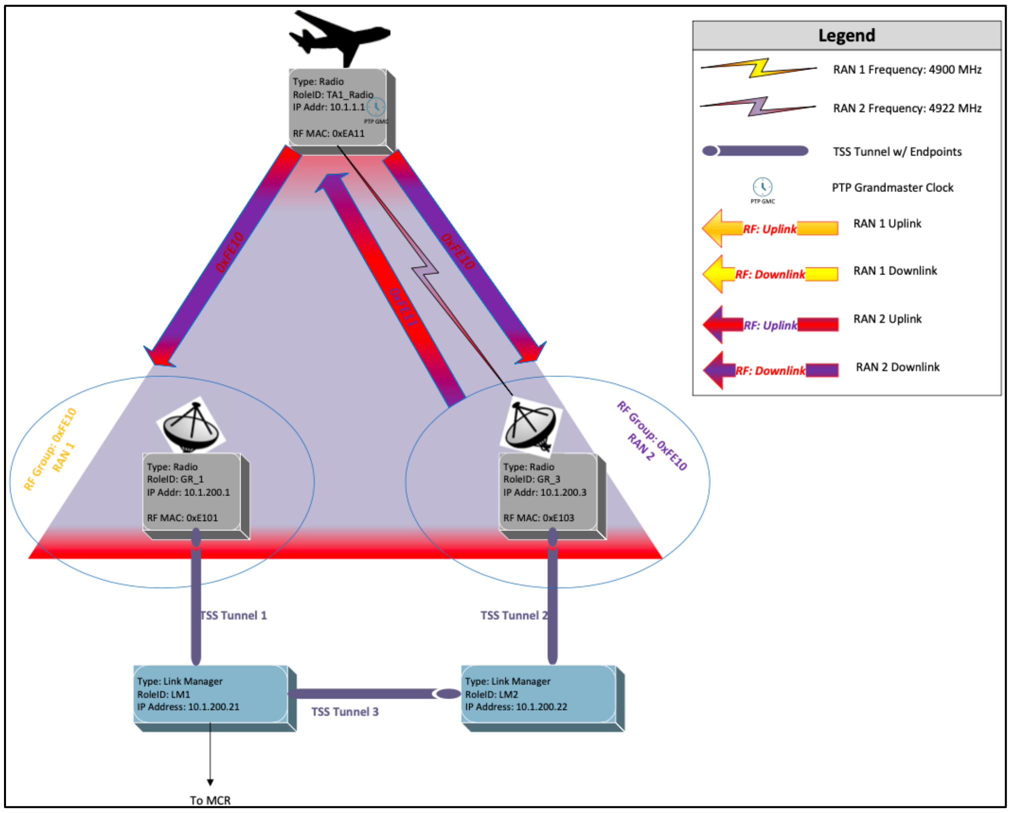


Figure 4-11. N2N Handoff: Post-Handoff State

When the N2N handoff process begins, the original LM, LM1 in our example, establishes a TSS tunnel with the LM of the new RF network to be handed off to. Uplink data begins to be routed from the MCR through LM1 and out the TSS tunnel towards LM2. While this route is changed, GR_1 is able to drain its uplink queue of data. Once complete, LM1 provides TA1_Radio with a transmission schedule that includes the new frequency. This transmission opportunity is the action that results in the TA1_Radio to begin transmitting and receiving on the new frequency of RAN 2. As the TA1_Radio transitions into RAN 2, LM2 is able to establish a connection with TA1_Radio through GR_3. Once connected, the dynamic transmission schedule is provided by LM2 for the TA1_Radio and GR_3 radios over the RAN 2 frequency. Network traffic from the TA1 network is routed from TA1_Radio to GR_3, through the TSS tunnel 2 connection to LM2, through the TSS tunnel 3 connection to LM1, and then on toward the MCR. This is depicted in Figure 4-11. Note that while GR_1 may still be in the transmission range of TA1_Radio after the N2N handoff, it does not receive any of the transmissions because it is tuned to only receive traffic from the RAN 1 frequency.

4.1.3.3 Range-to-Range (R2R) Handoff

An R2R handoff is very similar to the N2N handoff in that it involves transitioning the TA-based TmNS radio from one RF network to another. The real difference here, as its name implies, is that the handoff is from one test range to another. Thus, the new LM and new ground-based TmNS radios being handed off to are assets that are operated by a different range. Thus, some level of inter-range coordination is involved with the R2R handoff.

This type of handoff is needed whenever a test mission involves flights that utilize multiple test ranges. Exact implementations of R2R communication is dependent upon the ranges themselves. This type of handoff may be applicable to those ranges within relative close proximity of one another, such as some of those within the Southwest Range Complex, for instance.

For an R2R handoff, the handoff frequency of the TA-based TmNS radio should be set to that of the RAN frequency of the range that the TA is being handed off to. Different from the N2N handoff scenario, this could in fact be the same RAN frequency as is initially being used prior to the handoff. The original range 1 transmission schedule would simply be removed as part of the R2R handoff process. Depending on the proximity of the ranges involved in the R2R handoff event, there could be a significant amount of time post-handoff before the TA is in view of the antennas of range 2. For ranges that are in close proximity such that their coverage areas overlap, it is expected that the ranges will operate with different RAN frequencies. Under such circumstances, the R2R handoff will also be an N2N handoff due to the frequency change, but it would still include the inter-range communication portion associated with the R2R handoff.

The mechanism by which the inter-range communication is established has not been standardized by the TmNS. Because this is a range-specific detail, the connection details are left up to the ranges to resolve. This also includes the sharing of component configuration details. For instance, range 1 operators may not be granted authority to reconfigure a TmNS radio on range 2 for use in the inter-range test mission, but they can provide an MDL file to range 2 operators who can configure the appropriate TmNS radio for the incoming TA from range 1.

It is envisioned that both range 1 and range 2 will operate with their own SysMgr and LM applications.

4.2 Statically Allocated Transmission Schedules

There are two approaches to allocating transmission schedules for a RAN: static allocations and dynamic allocations. This section will focus on the first approach. Dynamic allocations are discussed later in Section 4.3.

Two-way telemetry links that utilize statically allocated transmission schedules do not require the use of an LM. The pre-allocated transmission schedules are provided to each TmNS radio during configuration through their MDL configuration file. The MDL file will contain a list of the epoch-based transmission schedule on a per-link basis.

The TmNS components required to provide the two-way telemetry capability include the following.

- Radio
- Network Switch
- System Manager

In the example network topology, RAN2 does not contain an LM. Instead, its RF links have statically allocated transmission schedules. These allocations are defined in the MDL configuration file. The example MDL configuration file can be found here.

Figure 4-12 shows the portion of the example topology used for static scheduling of RF links. It includes one ground-based TmNS radio and one airborne TmNS radio. The radios in the example are making use of RF multicast group addresses for transmission destinations.

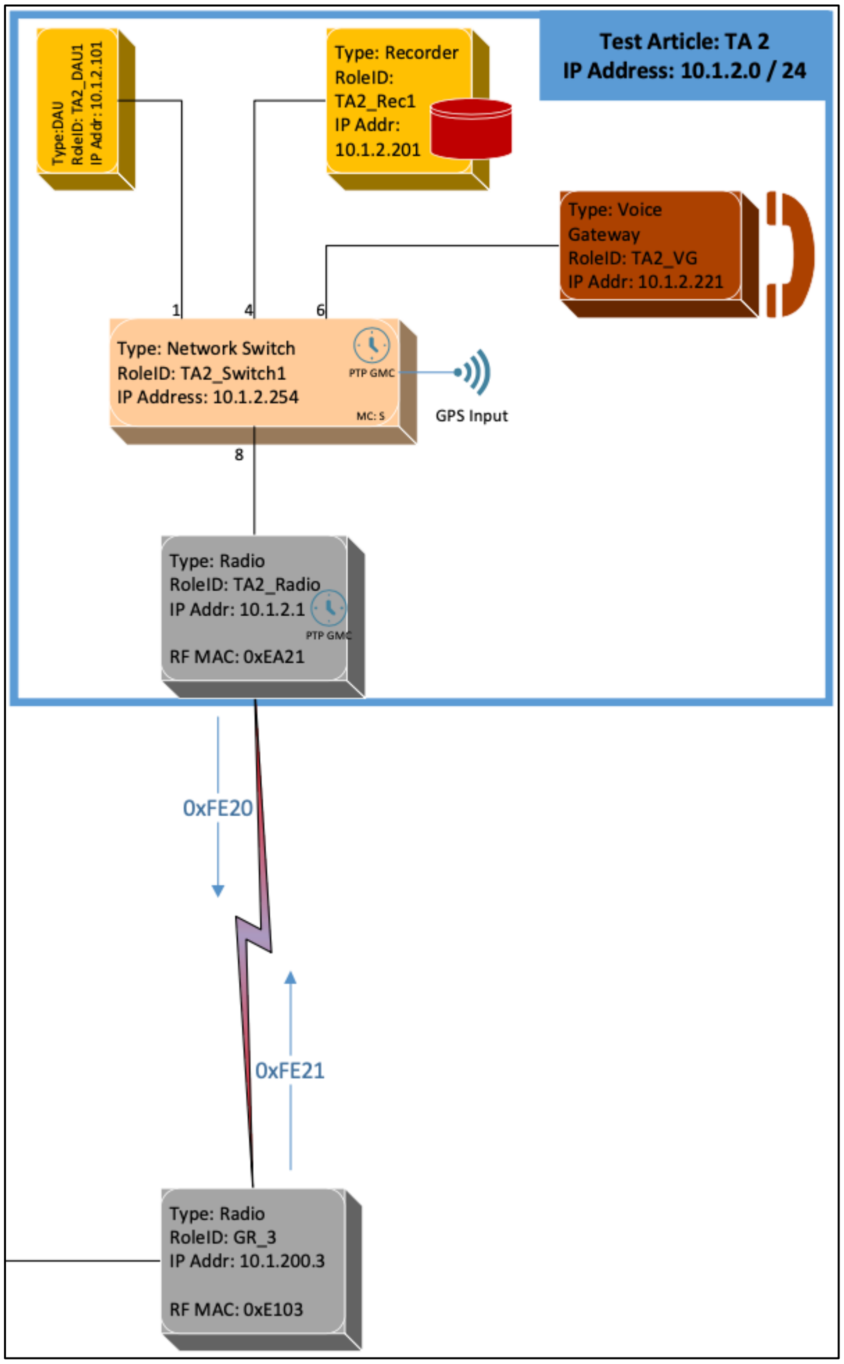


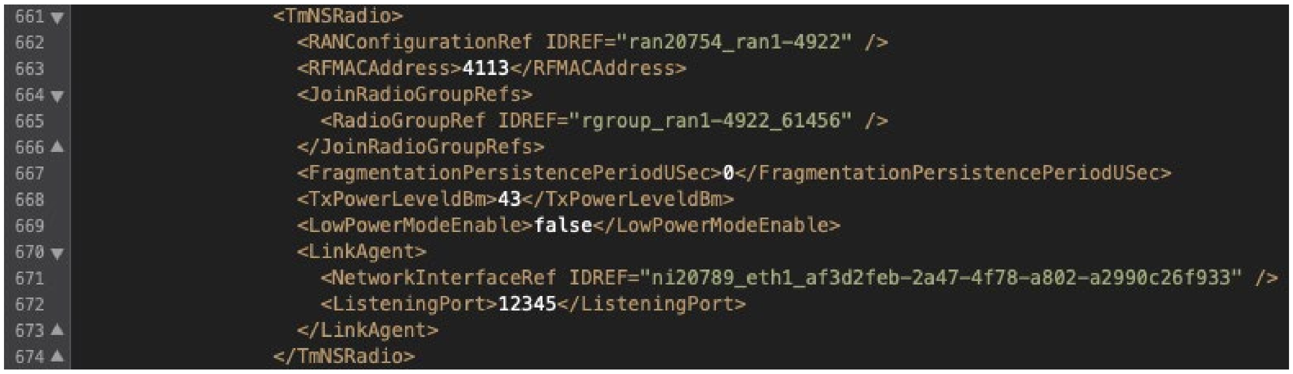
Figure 4-12. Example Topology for Static Link Scheduling

4.2.1 TmNS Radios

A pair of TmNS radios is needed in order to establish two-way telemetry communications. Through the MDL configuration file, each of the radios will be provided an RF MAC address by which it will identify its RF transmissions. The MDL configuration file will also identify the destination RF MAC address that each radio will transmit to. The RF links comprised of the source and destination RF MAC address combinations are specified along with the list of the statically allocated transmission opportunities for each link. The transmission opportunities specified are relative to their position within the epoch structure, and they repeat each epoch.

As has been discussed in a previous section, RF links can be either point-to-multipoint or point-to-point. The recommended approach is to utilize the point-to-multipoint RF links by providing group RF MAC addresses for the link destinations, and the example provided illustrates this approach.

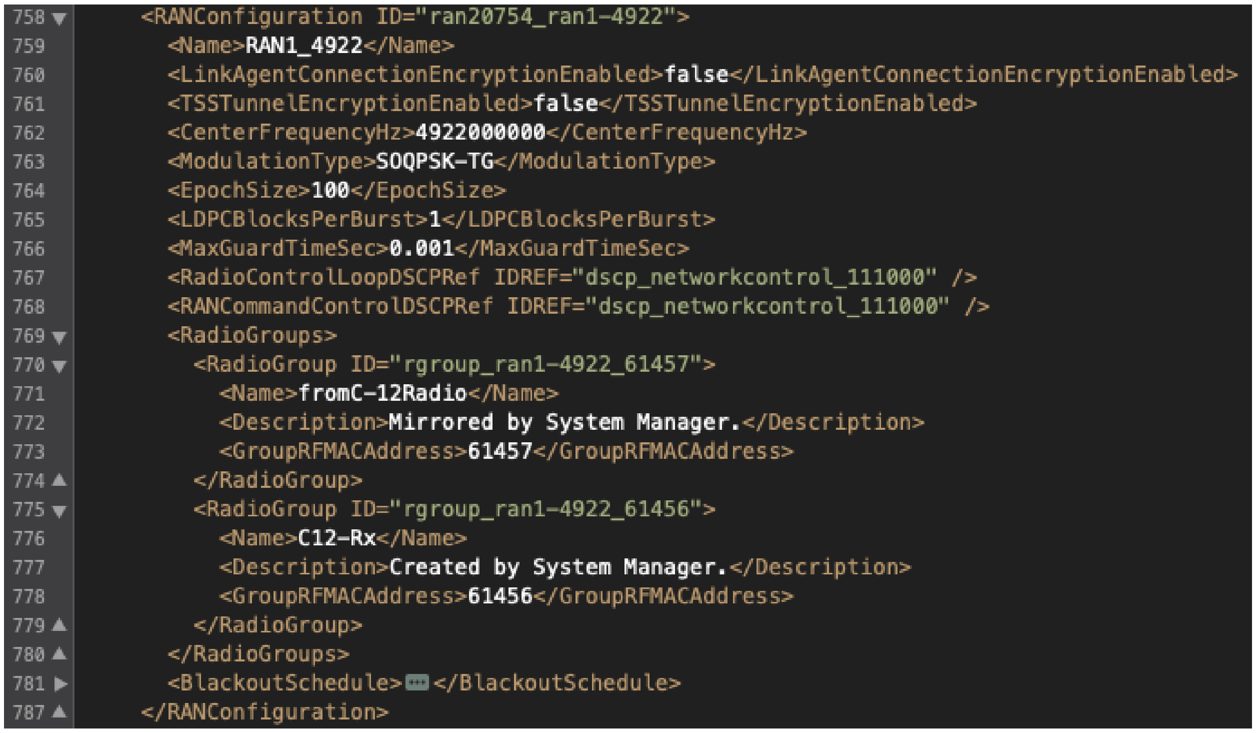
A portion of one radio's MDL configuration is shown in Figure 4-13. Notable elements include the <RFMACAddress> and the <JoinRadioGroupRef> elements. The <RFMACAddress> is specified with a decimal value, though the example figure displays the value in hexadecimal. The <JoinRadioGroupRef> element provides a reference to the group RF MAC address that the radio is subscribing to. The referenced radio group is defined in the RAN configuration pointed to by this radio through its <RANConfigurationRef>.



```
<TmNSRadio>
  <RANConfigurationRef IDREF="ran20754_ran1-4922" />
  <RFMACAddress>4113</RFMACAddress>
  <JoinRadioGroupRefs>
    <RadioGroupRef IDREF="rgroup_ran1-4922_61456" />
  </JoinRadioGroupRefs>
  <FragmentationPersistencePeriodUSec>0</FragmentationPersistencePeriodUSec>
  <TxPowerLeveldBm>43</TxPowerLeveldBm>
  <LowPowerModeEnable>false</LowPowerModeEnable>
  <LinkAgent>
    <NetworkInterfaceRef IDREF="ni20789_eth1_af3d2feb-2a47-4f78-a802-a2990c26f933" />
    <ListeningPort>12345</ListeningPort>
  </LinkAgent>
</TmNSRadio>
```

Figure 4-13. MDL for TmNSRadio Capability

The <RANConfiguration> for this example is shown in Figure 4-14. The <EpochSize> value (in milliseconds) sets the reference for valid static transmission assignments. The <RadioGroup> elements provide the group RF MAC address value that radios may subscribe to via their <JoinRadioGroupRef> element.



```
    <RANConfiguration ID="ran20754_ran1-4922">
      <Name>RAN1_4922</Name>
      <LinkAgentConnectionEncryptionEnabled>false</LinkAgentConnectionEncryptionEnabled>
      <TSSTunnelEncryptionEnabled>false</TSSTunnelEncryptionEnabled>
      <CenterFrequencyHz>4922000000</CenterFrequencyHz>
      <ModulationType>SOQPSK-TG</ModulationType>
      <EpochSize>100</EpochSize>
      <LDPCBlocksPerBurst>1</LDPCBlocksPerBurst>
      <MaxGuardTimeSec>0.001</MaxGuardTimeSec>
      <RadioControlLoopDSCPRef IDREF="dscp_networkcontrol_111000" />
      <RANCommandControlDSCPRef IDREF="dscp_networkcontrol_111000" />
      <RadioGroups>
        <RadioGroup ID="rgroup_ran1-4922_61457">
          <Name>fromC-12Radio</Name>
          <Description>Mirrored by System Manager.</Description>
          <GroupRFMACAddress>61457</GroupRFMACAddress>
        </RadioGroup>
        <RadioGroup ID="rgroup_ran1-4922_61456">
          <Name>C12-Rx</Name>
          <Description>Created by System Manager.</Description>
          <GroupRFMACAddress>61456</GroupRFMACAddress>
        </RadioGroup>
      </RadioGroups>
      <BlackoutSchedule>…</BlackoutSchedule>
    </RANConfiguration>
```

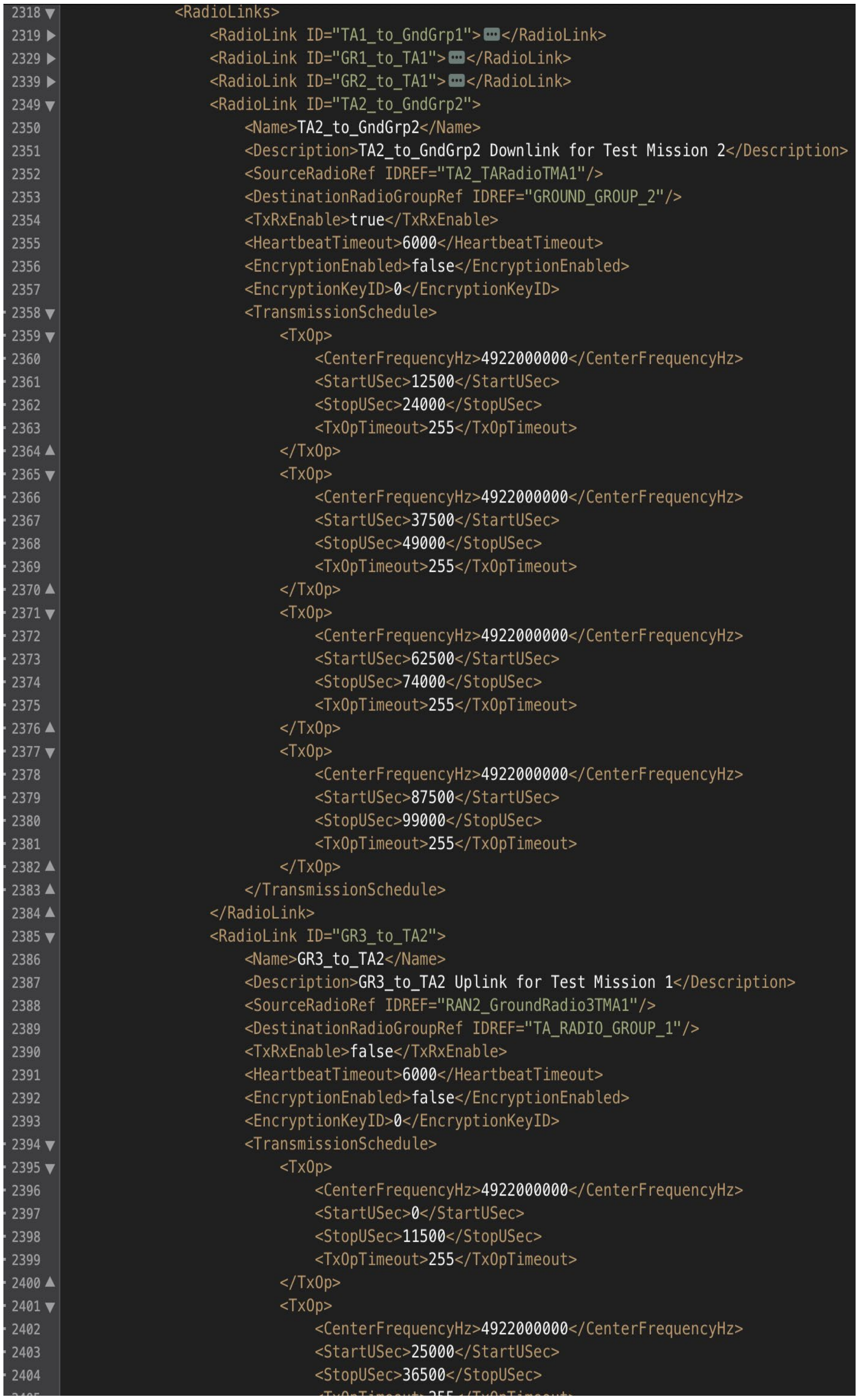
Figure 4-14. MDL for the RAN Configuration

The previously mentioned example MDL file, here, contains a TmNS radio and a RAN configuration.

4.2.1.1 Link Configuration

Defining RF links should follow the guidance specified in Subsection 4.1.2.2.3 for the best practices associated with defining RF links for a test mission. Unless a specific scenario warrants, links should utilize group RF MAC addresses for link destinations.

In addition to defining the link destination group RF MAC addresses for the links, the transmission schedules must also be provided in the MDL configuration file for each link. The link definitions in an MDL file provide references to both endpoints, from transmitting source radio to the destination group. The allocated transmission schedule for the link is provided as a list of <TxOp> elements that specify the start and stop time, in microseconds, of each transmission window relative to each epoch. A <TxOpTimeout> value of 255, the maximum value, is required to make the allocations infinite, never expiring. A portion of the MDL file that contains these RF link definitions is provided in Figure 4-15.



```
<RadioLinks>
    <RadioLink ID="TA1_to_GndGrp1">⋯</RadioLink>
    <RadioLink ID="GR1_to_TA1">⋯</RadioLink>
    <RadioLink ID="GR2_to_TA1">⋯</RadioLink>
    <RadioLink ID="TA2_to_GndGrp2">
        <Name>TA2_to_GndGrp2</Name>
        <Description>TA2_to_GndGrp2 Downlink for Test Mission 2</Description>
        <SourceRadioRef IDREF="TA2_TARadioTMA1"/>
        <DestinationRadioGroupRef IDREF="GROUND_GROUP_2"/>
        <TxRxEnable>true</TxRxEnable>
        <HeartbeatTimeout>6000</HeartbeatTimeout>
        <EncryptionEnabled>false</EncryptionEnabled>
        <EncryptionKeyID>0</EncryptionKeyID>
        <TransmissionSchedule>
            <TxOp>
                <CenterFrequencyHz>4922000000</CenterFrequencyHz>
                <StartUSec>12500</StartUSec>
                <StopUSec>24000</StopUSec>
                <TxOpTimeout>255</TxOpTimeout>
            </TxOp>
            <TxOp>
                <CenterFrequencyHz>4922000000</CenterFrequencyHz>
                <StartUSec>37500</StartUSec>
                <StopUSec>49000</StopUSec>
                <TxOpTimeout>255</TxOpTimeout>
            </TxOp>
            <TxOp>
                <CenterFrequencyHz>4922000000</CenterFrequencyHz>
                <StartUSec>62500</StartUSec>
                <StopUSec>74000</StopUSec>
                <TxOpTimeout>255</TxOpTimeout>
            </TxOp>
            <TxOp>
                <CenterFrequencyHz>4922000000</CenterFrequencyHz>
                <StartUSec>87500</StartUSec>
                <StopUSec>99000</StopUSec>
                <TxOpTimeout>255</TxOpTimeout>
            </TxOp>
        </TransmissionSchedule>
    </RadioLink>
    <RadioLink ID="GR3_to_TA2">
        <Name>GR3_to_TA2</Name>
        <Description>GR3_to_TA2 Uplink for Test Mission 1</Description>
        <SourceRadioRef IDREF="RAN2_GroundRadio3TMA1"/>
        <DestinationRadioGroupRef IDREF="TA_RADIO_GROUP_1"/>
        <TxRxEnable>false</TxRxEnable>
        <HeartbeatTimeout>6000</HeartbeatTimeout>
        <EncryptionEnabled>false</EncryptionEnabled>
        <EncryptionKeyID>0</EncryptionKeyID>
        <TransmissionSchedule>
            <TxOp>
                <CenterFrequencyHz>4922000000</CenterFrequencyHz>
                <StartUSec>0</StartUSec>
                <StopUSec>11500</StopUSec>
                <TxOpTimeout>255</TxOpTimeout>
            </TxOp>
            <TxOp>
                <CenterFrequencyHz>4922000000</CenterFrequencyHz>
                <StartUSec>25000</StartUSec>
                <StopUSec>36500</StopUSec>
```

Figure 4-15. MDL Description of RF Radio Links with Static Transmission Schedule Allocations

4.2.1.2 QoS Policy

The QoS policies in the TmNS are associated with radio links. Because the RF interfaces represent the network throughput bottleneck due to the bandwidth speed compared to the wired network infrastructure in the system, QoS policies are needed in order to ensure the network is able to remain functioning whenever congestion on the RF network is encountered. The available RF spectrum is shared amongst all test missions operating within the same RAN. As a general rule, a RAN may be able to sustain about 11 megabits per second of network traffic.

The MDL configuration file describes the QoS policies. These policies define the bandwidth requirements for a particular class of traffic passing through the radio and being transmitted over its RF interface.

At a minimum, traffic class should be defined to classify the network traffic associated with RFNMs, which are associated with assigning and maintaining a dynamic transmission schedule across the RAN. These packets should be given the highest priority in order to ensure that they can be delivered ahead of other general network traffic.

When VoIP traffic is required for a test mission, a QoS policy that guarantees low-latency delivery needs to be provided in order to ensure that VoIP traffic is delivered in a timely manner.

Without any QoS policy defined, all traffic will be queued and dequeued on the radio like a first-in-first-out, meaning higher-priority traffic can get delayed or dropped due to an abundance of low-priority traffic.

The SysMgr application contains a few pre-built QoS policies. It is recommended to use one of these policies as a starting point. These policies can be duplicated, and then the values can be modified as necessary in order to customize the allocated bandwidth limits for a particular link of a test mission.

Ground-based radios and the associated airborne radios may have different QoS policies. This allows airborne transmissions to use higher data rates than the ground-based radios.

4.2.2 Network Configuration and Time Synchronization

The TmNS radios require tight time synchronization in order to utilize their allocated transmission opportunities. This can be accomplished through either a direct GPS connection to each radio, or it can come by way of IEEE 1588-2008 (PTP) packets over the wired network segment. If utilizing PTP over the network, then it is crucial that the network switches in the path between the PTP grandmaster and the radio support the IEEE 1588-2008 protocol in order to provide a path for sub-microsecond synchronization on the radios.

In the examples provided, the TmNS radios used on the airborne TA synchronize their internal clocks directly to GPS by way of a dedicated GPS signal input into the radio. However, the TmNS radios used on the ground-based networks utilize IEEE 1588-2008 for time synchronization. They indirectly synchronize to GPS time through the PTP GMC on the local ground network segment, which is locked to a GPS signal. In this environment, the network path between the PTP GMC and the radio needs to be fully PTP-aware in order to support the required timing accuracies in synchronization needed.

For either fixed or mobile antenna sites, a typical ground-based installation is likely to contain the TmNS radio with a network switch and a PTP GMC. There may be other components on the network as well, including other switches. Connecting the TmNS radio and the PTP GMC device to the same network switch would only require one switch to support the IEEE 1588-2008 protocol. The PTP-aware switches can support the IEEE 1588-2008 protocol through either a boundary clock approach or a transparent clock approach.

Because IEEE 1588-2008 support is not a unique TmNS-specific capability, non-TmNS network switches can be used on the ground network as part of the transport path between the PTP GMC and the TmNS radio. However, the configuration of a non-TmNS switch is out of scope of the SysMgr and will need to be handled out-of-band.

Across a test range, it is likely that different antenna sites will have different IP spaces separated by routers. For each IP space, a different PTP GMC should be provided. Sites with multiple antennas and radios that are connected on the same network segment (e.g., within the same IP space) can use the same PTP GMC, but the same requirement for a fully PTP-aware network path between grandmaster and radio still applies.

4.2.3 System Manager

For the two-way telemetry capability, the SysMgr application provides a GUI for the user to be able to generate the MDL configuration files. The GUI allows for the RF link associations to be made between radios, and it provides a scheduling utility for visualizing and defining the static transmission opportunities per link. When creating static schedules for the RAN, the user is responsible for ensuring that transmission assignments for different links do not overlap within the epoch, that latency requirements can be met, and that bandwidth requirements are being met. The schedule visualization is provided to aid the user in generating link schedules that satisfy all the required constraints.

The SysMgr application may also be used during flight test operations to perform live health and status monitoring of the TmNS radios. This is not required but may be useful to have available for situational awareness or network troubleshooting.

4.3 Dynamically Allocated Transmission Schedules

Two-way telemetry links can be dynamically allocated with the use of an LM component in the RAN. By relying on an LM for generating the transmission schedules, the radio configuration is simplified in that a static transmission schedule does not need to be explicitly created and managed by a user for each radio and link in their respective MDL configuration files.

In addition to the ability to change transmission allocations of established links given current bandwidth demands across a RAN, the presence of an LM operating within the RAN opens the door for radio handoff scenarios.

The TmNS components required to provide the two-way telemetry capability with dynamically allocated transmission schedules include the following.

- Radio
- Link Manager
- Network Switch
- System Manager

The previously mentioned example MDL file, here, contains an LM, radios, and a RAN that supports dynamically allocated transmission schedules for its links.

Figure 4-16 shows the portion of the example topology used for dynamic scheduling for its RF links. It includes an LM and three TmNS radios. These radios use RF multicast group addresses for transmission destinations. The same transmission from TA1_Radio can be seen by both of the other two radios.

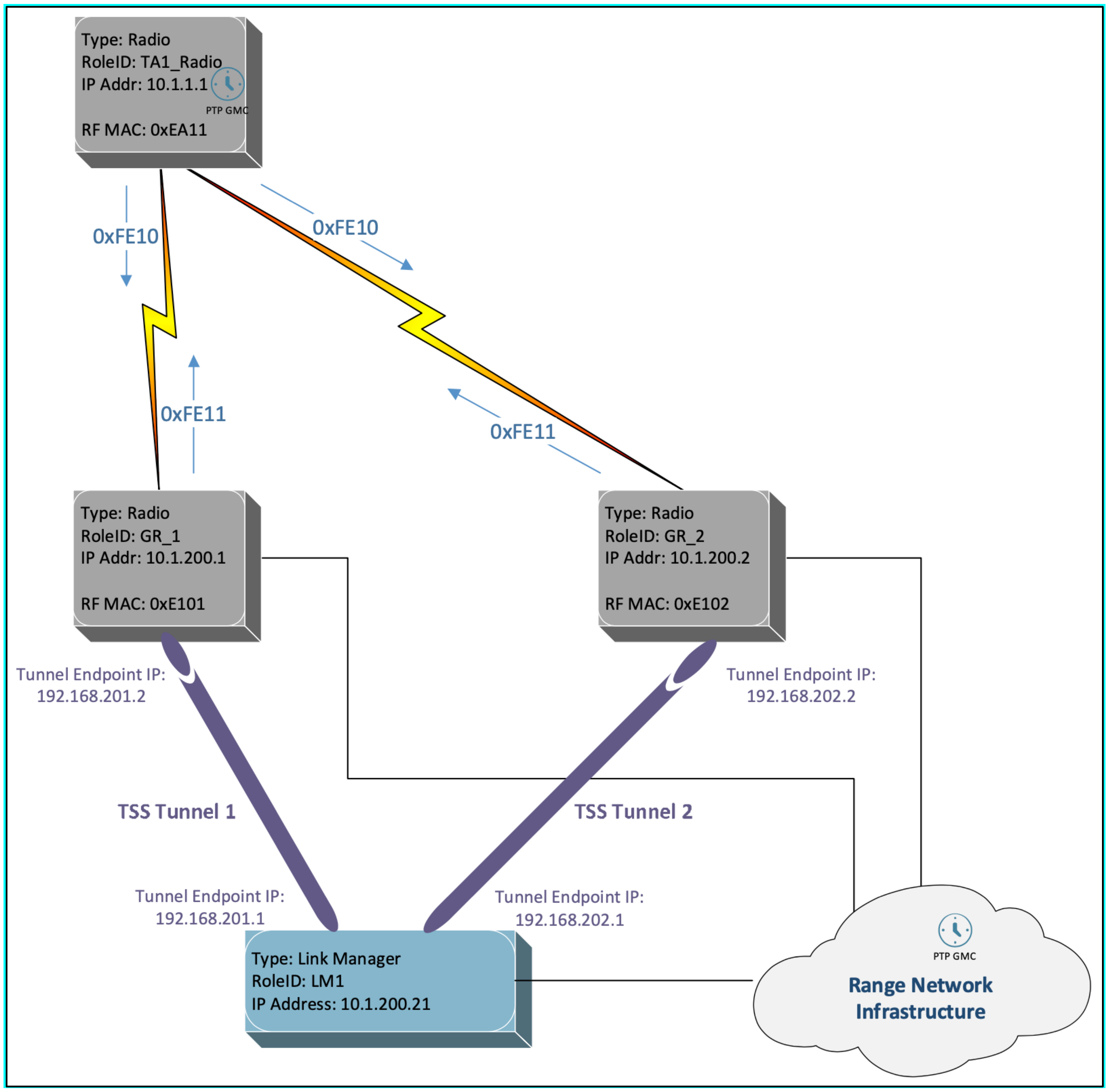


Figure 4-16. Example Topology for Dynamic Link Scheduling in a Radio Handoff Environment

4.3.1 TmNS Radios

Even for dynamically scheduled links, a minimum of two TmNS radios are required for establishing the two-way telemetry channel. When the RF links are dynamically scheduled, RF multicast allows additional TmNS radios to be easily added to the channel in support of handoff capabilities. In the example provided, there are three TmNS radios in the topology - one airborne TmNS radio and two ground-based TmNS radios.

Unlike the statically scheduled radios described previously, the MDL configuration for each of these TmNS radios does not need to specify any transmission opportunities within the file itself. They will not transmit onto the RF network until the LM connects to them and provides the transmission schedules for their links.

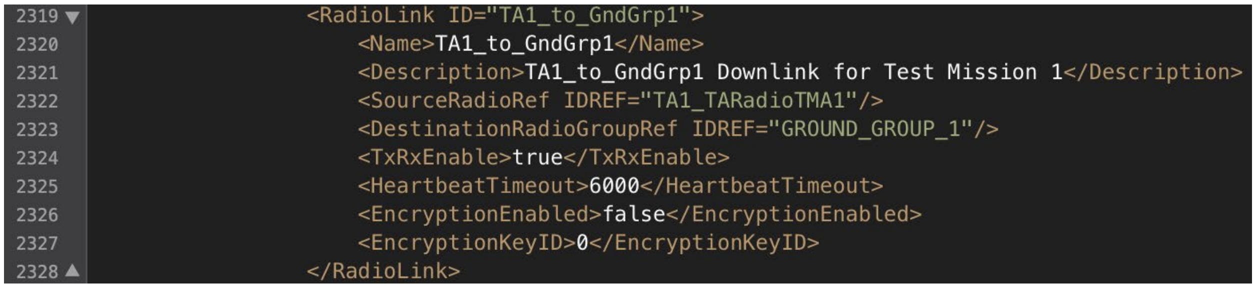
The TmNS radio is configured to support one RAN at a time. As such, its configuration MDL file provides a reference to a single RAN configuration.

4.3.1.1 Link Configuration

Definitions of RF links should utilize point-to-multipoint links via destination group RF MAC addresses. This is of particular importance for transmissions from a single link being received by multiple radios, such as when transmissions from an airborne radio need to be received by multiple ground-based radios. This approach also allows additional ground-based radios to be easily added to the radio group on the ground throughout the course of a flight test, such as when new ground radio assets become available during the test mission.

Proper link configuration is crucial in order to be able to perform handoff events with little or no interruption in communication. See Subsection 4.1.2.2.3 regarding best practices and guidance for defining RF links.

Link definitions may not have any transmission opportunities defined. When none are defined, a radio will not transmit until after it receives a schedule from the LM. Figure 4-17 shows the MDL description of a <RadioLink> that does not have an associated <TransmissionSchedule>.


```
<RadioLink ID="TA1_to_GndGrp1">
    <Name>TA1_to_GndGrp1</Name>
    <Description>TA1_to_GndGrp1 Downlink for Test Mission 1</Description>
    <SourceRadioRef IDREF="TA1_TARadioTMA1"/>
    <DestinationRadioGroupRef IDREF="GROUND_GROUP_1"/>
    <TxRxEnable>true</TxRxEnable>
    <HeartbeatTimeout>6000</HeartbeatTimeout>
    <EncryptionEnabled>false</EncryptionEnabled>
    <EncryptionKeyID>0</EncryptionKeyID>
</RadioLink>
```
Figure 4-17. MDL Example Description of a Radio Link for Dynamic Transmission Schedule Allocations

4.3.1.2 QoS Policy

The QoS policies used by TmNS radios in a dynamically scheduled RAN still follow the same guidance as stated for a statically scheduled RAN in Subsection 4.2.1.2. Additionally, the LM can use the bandwidth limits per policy used by a link as part of its scheduling allocation algorithm. It would be wasteful of the available bandwidth to allocate more transmission capacity to a radio in excess of its transmission link limits. This is important whenever the LM provides active transmission schedules for multiple test missions.

4.3.1.3 TSS Tunnels on a Radio

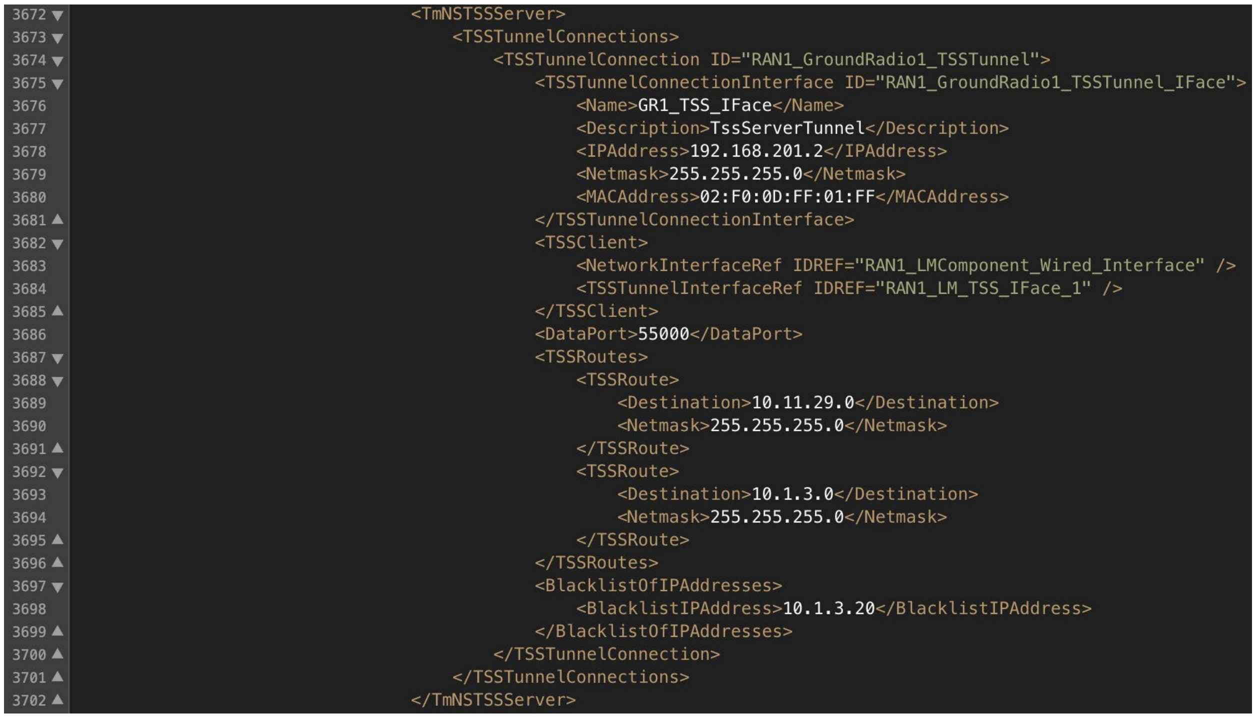
A TmNS radio may be configured to operate as a TSS server that listens for incoming TSS tunnel connections from an LM. When a connection is established, a new virtual interface is activated on the radio, and routing rules obtained from the MDL configuration file are applied. This will enable routing of data received from the wireless RF interface to be placed into a TSS Data Message and delivered through the TSS tunnel to the LM.

The TSS Data Messages received by the TmNS radio from the TSS tunnel are processed to extract the encapsulated Ethernet frame, and then the frame is routed to its next hop interface, which is likely to be its wireless RF interface.

The TmNS radios may not always have a TSS server active. This is determined by the MDL configuration file. If there is no TSS server capability defined, then the radio will not listen for incoming TSS tunnel connections.

The TSS server capability is only utilized on TmNS radios located on the ground-based network. The TSS server capability is only required whenever there are multiple ground-based radios operating as part of the same test mission. The TmNS radios on airborne platforms do not operate as TSS servers.

The MDL description of a TSS server contains the virtual network interface definition, the expected TSS client interface reference, and any routes that are associated with the interface. Any IP addresses listed within the list of blacklisted IP addresses are prevented from being used within the tunnel. The blacklist must contain the IP address of the TSS client that connects to the TSS server. Otherwise, the tunnel will fail to properly function. An example of the MDL configuration for a TSS server can be found in Figure 4-18.



```xml
<TmNSTSSServer>
    <TSSTunnelConnections>
        <TSSTunnelConnection ID="RAN1_GroundRadio1_TSSTunnel">
            <TSSTunnelConnectionInterface ID="RAN1_GroundRadio1_TSSTunnel_IFace">
                <Name>GR1_TSS_IFace</Name>
                <Description>TssServerTunnel</Description>
                <IPAddress>192.168.201.2</IPAddress>
                <Netmask>255.255.255.0</Netmask>
                <MACAddress>02:F0:0D:FF:01:FF</MACAddress>
            </TSSTunnelConnectionInterface>
            <TSSClient>
                <NetworkInterfaceRef IDREF="RAN1_LMComponent_Wired_Interface" />
                <TSSTunnelInterfaceRef IDREF="RAN1_LM_TSS_IFace_1" />
            </TSSClient>
            <DataPort>55000</DataPort>
            <TSSRoutes>
                <TSSRoute>
                    <Destination>10.11.29.0</Destination>
                    <Netmask>255.255.255.0</Netmask>
                </TSSRoute>
                <TSSRoute>
                    <Destination>10.1.3.0</Destination>
                    <Netmask>255.255.255.0</Netmask>
                </TSSRoute>
            </TSSRoutes>
            <BlacklistOfIPAddresses>
                <BlacklistIPAddress>10.1.3.20</BlacklistIPAddress>
            </BlacklistOfIPAddresses>
        </TSSTunnelConnection>
    </TSSTunnelConnections>
</TmNSTSSServer>
```

Figure 4-18. MDL Example Description of a <TmNSTSSServer>

4.3.2 Link Manager

A TmNS LM provides transmission schedules to TmNS radios during test operations in order to optimize bandwidth utilization across all active test missions operating over an RF network. A single LM is expected to manage all TmNS radio assets on a single two-way telemetry frequency. Generally speaking, there exists a one-to-one relationship between an LM and a RAN on a range.

While test missions are transient by nature, the operation of an LM is expected to be active throughout the entire day, continuing to operate as test missions are added and removed throughout the day. Thus, the LM component differs from other TmNS components in that it must continue to operate when being reconfigured. New MDL configuration files are provided to the LM to add or remove radios and links as test missions are added or removed from operation throughout the day on the range.

Adding an LM to the TmNS system adds more complexity, but there are certain benefits that can be leveraged by doing so. With an LM as part of the TmNS system, bandwidth utilization shared across multiple concurrent test missions can be optimized, providing more bandwidth to those links that can benefit from more transmission time, if available. An LM also brings capabilities to manage and perform various handoff scenarios.

An LM also performs an IPv4 routing function. This function works in conjunction with the use of the TSS tunnels connected on the LM in support of RF multicast and handoff operations. Specific details of its routing capability are described in Subsection 4.3.2.2.

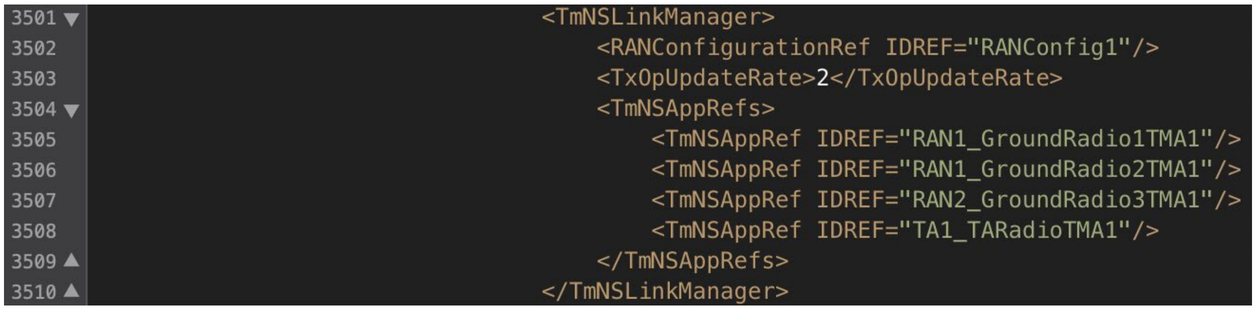
This section will include LM-specific configuration settings, TSS tunnels running on the LM, and the LM's role in the various handoff scenarios.

4.3.2.1 LM Configuration Settings

There are multiple sections of MDL that drive the configuration of the LM. These start within the definition of the LM's <NetworkNode>. From here, the <TmNSLinkManager> and the optional <TmNSTSSClient> and <TmNSTSSServer> elements are defined. These provide references by which the rest of the MDL configuration file can be searched. The relevant portions of MDL are highlighted in the following subsections.

4.3.2.1.1 <TmNSLinkManager> Branch of MDL Configuration

The first key area of the MDL configuration file is the <TmNSLinkManager> branch of the <NetworkNode> for the LM component. This branch requires a reference to the specific RAN configuration element in the MDL file that the LM is to operate according to. It also requires a value for the maximum number of epochs that the LM may wait before sending a new schedule update for the links that it manages. The LM also expects a list of TmNS application references in the MDL file that points to radios that have links that are to be managed by the LM. Figure 4-19 provides the relevant portion of MDL for the <TmNSLinkManager> element description for the example.



```
<TmNSLinkManager>
    <RANConfigurationRef IDREF="RANConfig1"/>
    <TxOpUpdateRate>2</TxOpUpdateRate>
    <TmNSAppRefs>
        <TmNSAppRef IDREF="RAN1_GroundRadio1TMA1"/>
        <TmNSAppRef IDREF="RAN1_GroundRadio2TMA1"/>
        <TmNSAppRef IDREF="RAN2_GroundRadio3TMA1"/>
        <TmNSAppRef IDREF="TA1_TARadioTMA1"/>
    </TmNSAppRefs>
</TmNSLinkManager>
```

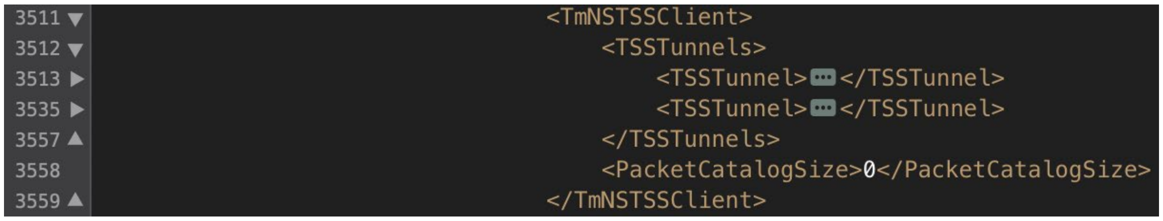
Figure 4-19. MDL Example Description of a <TmNSLinkManager> Element

Each <TmNSAppRef> element points to a radio that the LM is expected to provide dynamic schedule allocations. The LM parses each referenced component in order to find an associated IP address by which it can establish network connections to for conveying RFNMs, such as transmission schedule updates and queue status messages. The LM also parses the MDL for all of the <RadioLink> elements in the file as well. For each link that specifies a <SourceRadioRef> that matches one of its listed <TmNSAppRef> radios, the LM may be required to generate transmission schedules.

4.3.2.1.2 <TmNSTSSClient> and <TmNSTSSServer> Branches of MDL Configuration

Though optional in the MDL configuration file for an LM, the MDL configuration description may contain TSS tunnel client and server descriptions. When present in the file, they define both the local TSS tunnel endpoints and the remote endpoint of each TSS tunnel. The main difference between <TmNSTSSClient> and <TmNSTSSServer> element definitions is that TSS client descriptions imply that the LM will initiate the connection whereas TSS server descriptions imply that the TSS tunnel connection will be initiated by the associated TSS client.

The majority of the time, the LM will operate as a TSS client, and it will initiate connections to radios that operate as TSS servers. However, in support of R2R handoffs, TSS tunnel connections are made between two LMs; thus requiring one of them to operate as a TSS server. In the example, LM1 has two TSS tunnel endpoints defined. It is the TSS client side for both endpoints, meaning that the LM will initiate connections to the defined remote endpoints. This can be seen in Figure 4-20. The example does not have a <TmNSTSSServer> defined.



```
<TmNSTSSClient>
    <TSSTunnels>
        <TSSTunnel>⋯</TSSTunnel>
        <TSSTunnel>⋯</TSSTunnel>
    </TSSTunnels>
    <PacketCatalogSize>0</PacketCatalogSize>
</TmNSTSSClient>
```

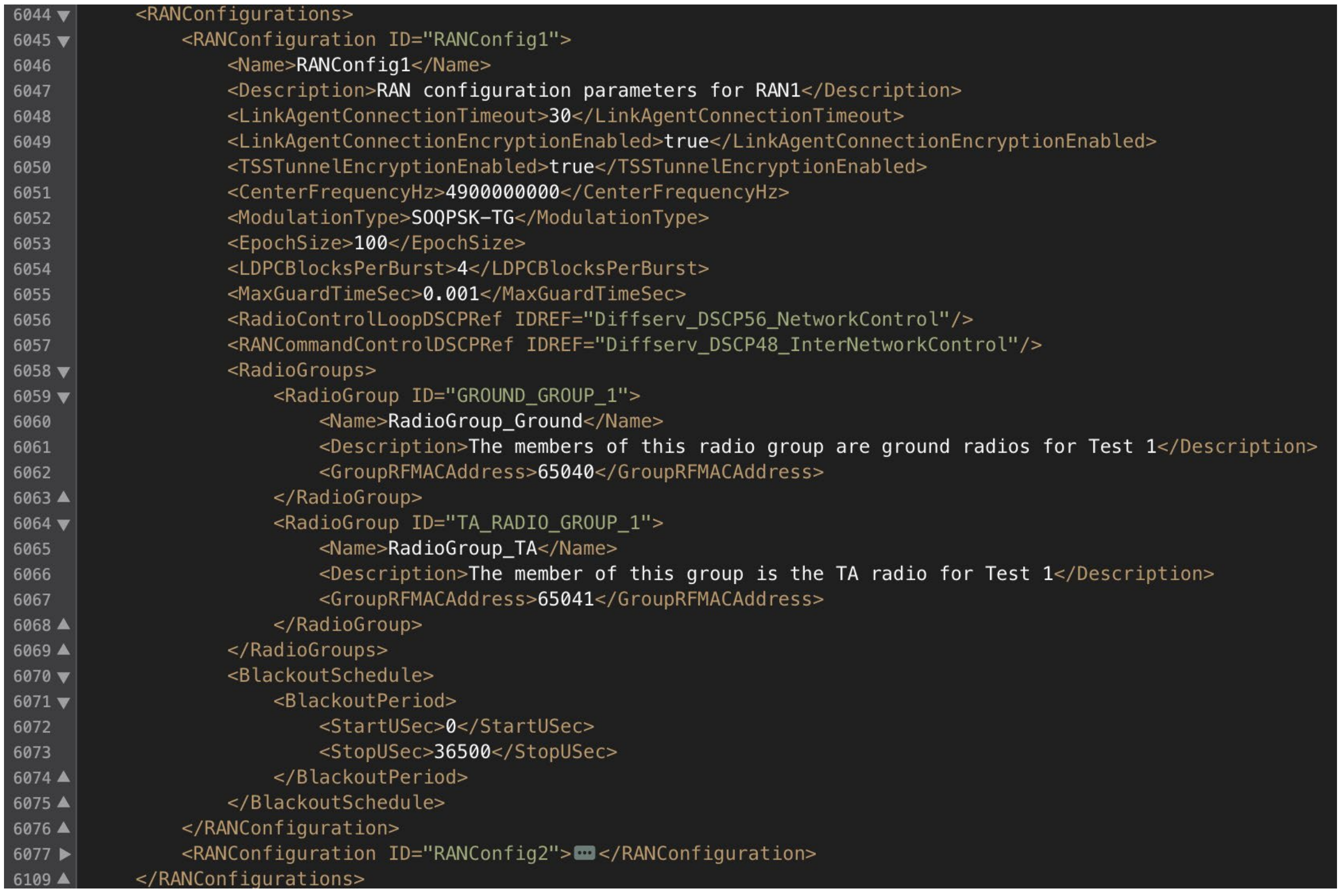
Figure 4-20. MDL Example Description of a <TmNSTSSClient> Element

The <PackageCatalogSize> refers to the number of packets the LM will store and use for detecting duplicate IP packets that it receives across all of its TSS tunnels. The larger the value, the greater the memory requirement and the time required to check for duplication. Too small of a value may result in duplicated packets going undetected through the system. Setting the value to 0 results in a default value being used by the LM according to a local configuration parameter that is out of scope of IRIG 106. The package catalog size is only relevant to the TSS tunnel client.

4.3.2.1.3 <RANConfiguration> for a Link Manager

The <TmNSLinkManager> contains a reference to a <RANConfiguration> in the MDL configuration file. This referenced <RANConfiguration> contains configuration parameters that impact the LM's scheduling algorithm, such as the epoch size and maximum guard time to use between TxOps from different radios. It also specifies the frequency of the RF network, which the LM specifies in each TxOp message that it provides to radios. Settings for whether LM connections for TxOp delivery and TSS tunnels are encrypted or not are also provided. There are also references to DSCP elements for radio and RAN control messages.

The <RANConfiguration> also includes <RadioGroup> descriptions that are used within the RAN. The configuration may also include a blackout schedule of one or more periods of time. If any are defined, the LM scheduling algorithm will not assign any TxOps for any links during these periods. These configuration parameters can all be seen in Figure 4-21.



```
<RANConfigurations>
    <RANConfiguration ID="RANConfig1">
        <Name>RANConfig1</Name>
        <Description>RAN configuration parameters for RAN1</Description>
        <LinkAgentConnectionTimeout>30</LinkAgentConnectionTimeout>
        <LinkAgentConnectionEncryptionEnabled>true</LinkAgentConnectionEncryptionEnabled>
        <TSSTunnelEncryptionEnabled>true</TSSTunnelEncryptionEnabled>
        <CenterFrequencyHz>4900000000</CenterFrequencyHz>
        <ModulationType>SOQPSK-TG</ModulationType>
        <EpochSize>100</EpochSize>
        <LDPCBlocksPerBurst>4</LDPCBlocksPerBurst>
        <MaxGuardTimeSec>0.001</MaxGuardTimeSec>
        <RadioControlLoopDSCPRef IDREF="Diffserv_DSCP56_NetworkControl"/>
        <RANCommandControlDSCPRef IDREF="Diffserv_DSCP48_InterNetworkControl"/>
        <RadioGroups>
            <RadioGroup ID="GROUND_GROUP_1">
                <Name>RadioGroup_Ground</Name>
                <Description>The members of this radio group are ground radios for Test 1</Description>
                <GroupRFMACAddress>65040</GroupRFMACAddress>
            </RadioGroup>
            <RadioGroup ID="TA_RADIO_GROUP_1">
                <Name>RadioGroup_TA</Name>
                <Description>The member of this group is the TA radio for Test 1</Description>
                <GroupRFMACAddress>65041</GroupRFMACAddress>
            </RadioGroup>
        </RadioGroups>
        <BlackoutSchedule>
            <BlackoutPeriod>
                <StartUSec>0</StartUSec>
                <StopUSec>36500</StopUSec>
            </BlackoutPeriod>
        </BlackoutSchedule>
    </RANConfiguration>
    <RANConfiguration ID="RANConfig2">⋯</RANConfiguration>
</RANConfigurations>
```

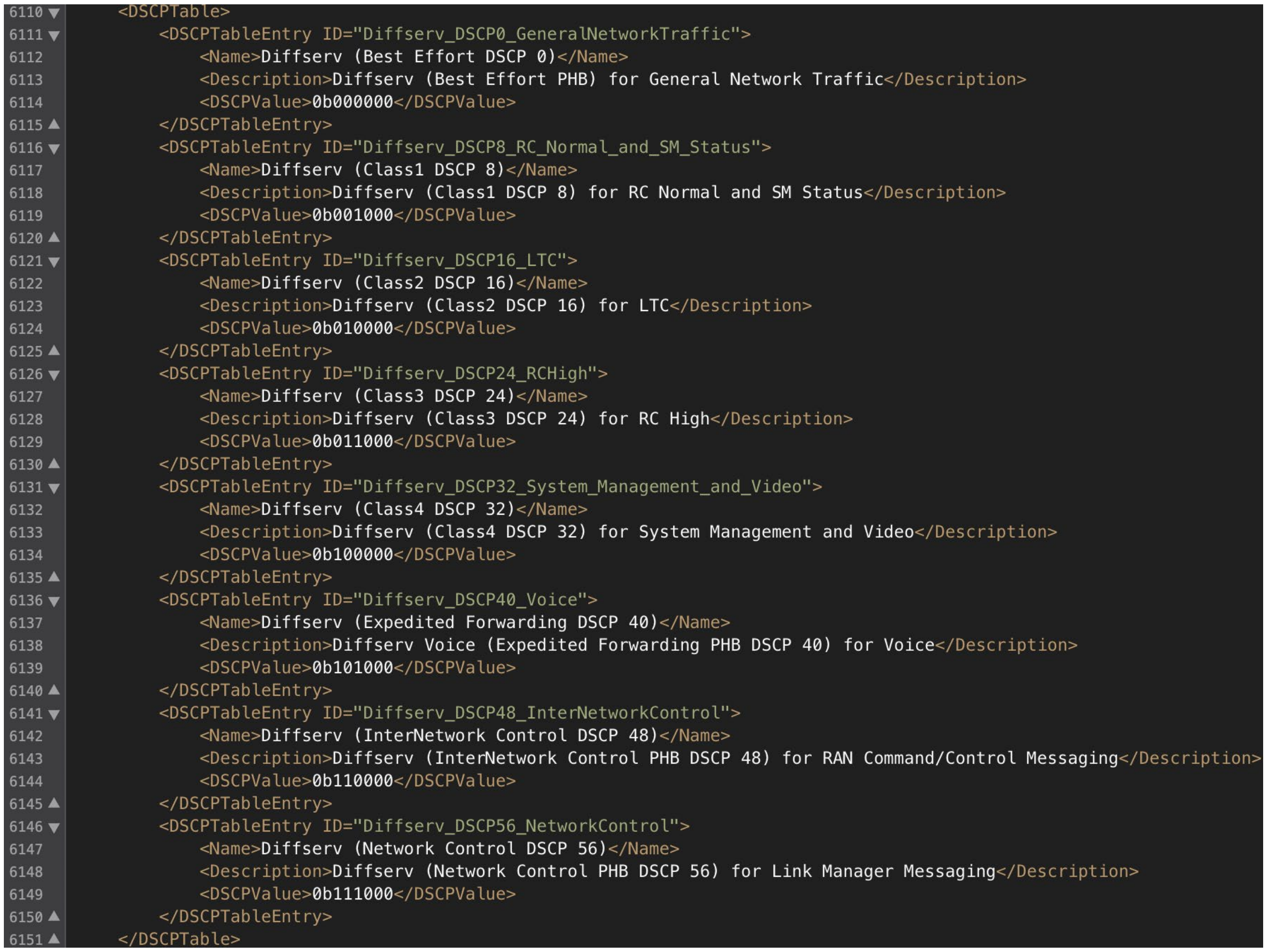
Figure 4-21. MDL Example Description of a <RANConfiguration> Element

4.3.2.1.4 <DSCPTable> References for a Link Manager

In order to ensure that RFNMs between the LM and radios are exchanged in a timely manner, the RFNM packets need to be marked with dedicated DSCP markings and utilized in the radios through appropriate QoS policies. The <RANConfiguration> specifies a <RadioControlLoopDSCPRef> and a <RANCommandControlDSCPRef>, both of which point to a <DSCPTableEntry> element within the <DSCPTable> portion of the MDL configuration file.

Both of these references need to point to entries that are used for higher-level network traffic. The QoS policies implemented on the radios should be described in MDL such that these traffic classes are given the highest priority in order to ensure that the control messages from the LM that contain transmission schedule updates are not queued in transit with other network traffic through the radios.

The MDL description of the DSCP table for the example can be seen in Figure 4-22. This example matches the guidance provided by Appendix 22-A.

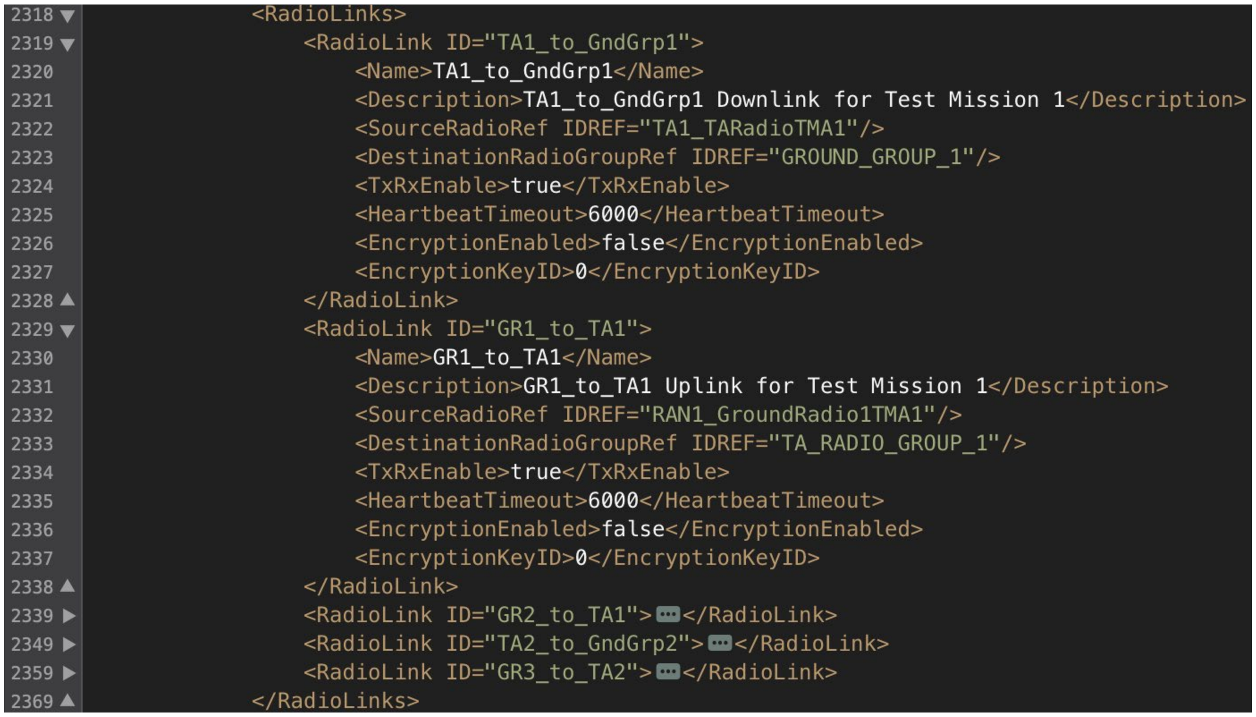


```
    <DSCPTable>
        <DSCPTableEntry ID="Diffserv_DSCP0_GeneralNetworkTraffic">
            <Name>Diffserv (Best Effort DSCP 0)</Name>
            <Description>Diffserv (Best Effort PHB) for General Network Traffic</Description>
            <DSCPValue>0b000000</DSCPValue>
        </DSCPTableEntry>
        <DSCPTableEntry ID="Diffserv_DSCP8_RC_Normal_and_SM_Status">
            <Name>Diffserv (Class1 DSCP 8)</Name>
            <Description>Diffserv (Class1 DSCP 8) for RC Normal and SM Status</Description>
            <DSCPValue>0b001000</DSCPValue>
        </DSCPTableEntry>
        <DSCPTableEntry ID="Diffserv_DSCP16_LTC">
            <Name>Diffserv (Class2 DSCP 16)</Name>
            <Description>Diffserv (Class2 DSCP 16) for LTC</Description>
            <DSCPValue>0b010000</DSCPValue>
        </DSCPTableEntry>
        <DSCPTableEntry ID="Diffserv_DSCP24_RCHigh">
            <Name>Diffserv (Class3 DSCP 24)</Name>
            <Description>Diffserv (Class3 DSCP 24) for RC High</Description>
            <DSCPValue>0b011000</DSCPValue>
        </DSCPTableEntry>
        <DSCPTableEntry ID="Diffserv_DSCP32_System_Management_and_Video">
            <Name>Diffserv (Class4 DSCP 32)</Name>
            <Description>Diffserv (Class4 DSCP 32) for System Management and Video</Description>
            <DSCPValue>0b100000</DSCPValue>
        </DSCPTableEntry>
        <DSCPTableEntry ID="Diffserv_DSCP40_Voice">
            <Name>Diffserv (Expedited Forwarding DSCP 40)</Name>
            <Description>Diffserv Voice (Expedited Forwarding PHB DSCP 40) for Voice</Description>
            <DSCPValue>0b101000</DSCPValue>
        </DSCPTableEntry>
        <DSCPTableEntry ID="Diffserv_DSCP48_InterNetworkControl">
            <Name>Diffserv (InterNetwork Control DSCP 48)</Name>
            <Description>Diffserv (InterNetwork Control PHB DSCP 48) for RAN Command/Control Messaging</Description>
            <DSCPValue>0b110000</DSCPValue>
        </DSCPTableEntry>
        <DSCPTableEntry ID="Diffserv_DSCP56_NetworkControl">
            <Name>Diffserv (Network Control DSCP 56)</Name>
            <Description>Diffserv (Network Control PHB DSCP 56) for Link Manager Messaging</Description>
            <DSCPValue>0b111000</DSCPValue>
        </DSCPTableEntry>
    </DSCPTable>
```

Figure 4-22. MDL Example Description of the DSCP Table

4.3.2.1.5 <RadioLinks> References for a Link Manager

The LM periodically runs its scheduling algorithm. Each time it does, it creates a non-overlapping transmission schedule for all links it is managing. Those radios in a receive-only mode at schedule creation time are not allocated any transmission opportunities. The links being managed by the LM are traced through the references to the <TmNSAppRefs> provided and the <RadioLinks> defined elsewhere in the MDL configuration file. The radio links that the LM may allocate transmission schedules to include any <RadioLink> that has a <SourceRadioRef> that points to one of the <TmNSAppRefs> that it manages. A portion of the <RadioLinks> of the example is provided in Figure 4-23.



```
        <RadioLinks>
            <RadioLink ID="TA1_to_GndGrp1">
                <Name>TA1_to_GndGrp1</Name>
                <Description>TA1_to_GndGrp1 Downlink for Test Mission 1</Description>
                <SourceRadioRef IDREF="TA1_TARadioTMA1"/>
                <DestinationRadioGroupRef IDREF="GROUND_GROUP_1"/>
                <TxRxEnable>true</TxRxEnable>
                <HeartbeatTimeout>6000</HeartbeatTimeout>
                <EncryptionEnabled>false</EncryptionEnabled>
                <EncryptionKeyID>0</EncryptionKeyID>
            </RadioLink>
            <RadioLink ID="GR1_to_TA1">
                <Name>GR1_to_TA1</Name>
                <Description>GR1_to_TA1 Uplink for Test Mission 1</Description>
                <SourceRadioRef IDREF="RAN1_GroundRadio1TMA1"/>
                <DestinationRadioGroupRef IDREF="TA_RADIO_GROUP_1"/>
                <TxRxEnable>true</TxRxEnable>
                <HeartbeatTimeout>6000</HeartbeatTimeout>
                <EncryptionEnabled>false</EncryptionEnabled>
                <EncryptionKeyID>0</EncryptionKeyID>
            </RadioLink>
            <RadioLink ID="GR2_to_TA1">…</RadioLink>
            <RadioLink ID="TA2_to_GndGrp2">…</RadioLink>
            <RadioLink ID="GR3_to_TA2">…</RadioLink>
        </RadioLinks>
```

Figure 4-23. MDL Example Description of <RadioLinks>

4.3.2.2 TSS Tunnels on an LM

The TSS tunnels on the LM serve two key purposes.

- Duplicate Packet Detection
- Tunnel Selection

Duplicate packet detection is associated with "downlink data", which is the network traffic traveling from an airborne radio down towards the LM. On the other hand, tunnel selection is associated with "uplink data", which is the network traffic that is being sent from the ground network up to the airborne network.

Duplicate packet detection occurs whenever multiple ground-based radios receive the same transmission from an airborne radio via an RF multicast link. Each ground-based radio routes the traffic through its TSS tunnel to the LM in the form of an TSS Data Message. When the LM receives a TSS Data Message, it searches its catalog of recent TSS Data Messages for a match. If no match is found, the encapsulated Ethernet message is routed to its next hop, and the message's CRC frame ID is cataloged for future reference. However, if a match is found, it is determined to be a duplicate, and the LM simply drops the encapsulated Ethernet message. The result is that only one copy of the messages transmitted from the airborne radio will be routed through the LM to the destination.

Tunnel selection is performed by the LM whenever it chooses a TSS tunnel for routing the uplink data to the airborne network. While there may be multiple TSS tunnels that are potential next hop routes to the airborne network, only one of the TSS tunnel remote endpoint radios is actively being allotted a dynamic transmission schedule by the LM at a time. Whichever ground-based radio is selected as the active uplink radio, the LM routes the uplink data into the associated TSS tunnel. During handoff events, the LM changes the TSS tunnel it uses for routing to the airborne network when it begins allocating the transmission schedule for the post-handoff radio.

AN LM can operate as either a TSS client or a TSS server. It is most common for the LM to operate as a TSS client and connect to the TSS server capability of the radios that it manages links for. To support N2N and R2R handoffs, an LM may run a TSS server in order to allow for an incoming TSS tunnel connection from another LM.

As a TSS client, the LM initiates tunnel connections to all TmNS radios that it manages links for. For each tunnel connection, the LM will create an associated virtual interface over which it can route traffic. In the example network, LM1 initiates two TSS tunnel connections, one to radio GR_1 and one to radio GR_2. TSS Tunnel 1 has an IP space of 192.168.201.0/24, and TSS Tunnel 2 has an IP space of 192.168.202.0/24.

As network traffic is transmitted from the airborne TA1_Radio and received by both ground-based radios GR_1 and GR_2, each of the ground radios will route the traffic through their respective TSS tunnel. The traffic to be routed is encapsulated in a TSS Data Message before being placed into the TSS tunnel. The first instance of the encapsulated Ethernet frame from the airborne transmission that is received by the LM is routed to its destination within MCR 1. The second instance of the frame that arrives will be detected as a duplicate and will be subsequently dropped.

As a TSS server, the LM listens for an incoming TSS tunnel connection from another LM. The TSS server runs on an LM that will represent the new LM in the N2N and R2R handoff scenarios. It still verifies that it does not receive any duplicated encapsulated Ethernet frames, though any messages it receives in this TSS tunnel will be uplink traffic destined for the airborne network, thus there should not be any other interfaces by which it receives the uplink data.

4.3.2.3 Link Manager's Role in Handoffs

The three different types of handoffs are described in detail in Subsection 4.1.3. This particular section will discuss the LM's role in the different handoff scenarios.

4.3.2.3.1 Link Manager's Role in A2A Handoffs

A single LM is required for A2A handoffs. Because of this, the LM is capable of performing automated handoffs according to a pre-defined set of rules to determine which ground-based TmNS radio to schedule for the uplink path at any given time. The SysMgr user can authorize or deauthorize the LM from performing the automated handoffs. Manual handoffs can still be initiated by a SysMgr user regardless of the authorization state of automated handoffs.

The LM initiates a TSS tunnel connection to each of the ground-based TmNS radios associated with a test mission. The LM operates as an IPv4 router, and it associates a route for the test mission's airborne networks with one of these tunnels. At any time during the test mission, the LM will only provide a transmission schedule to one ground-based TmNS radio. During the A2A handoff event, the LM performs a route change to the airborne networks, redirecting the network traffic through a different TSS tunnel whose endpoint is on the TmNS radio that is about to be handed off to. The LM allows the initial ground-based radio to empty its queue of data to be transmitted to the TA networks, and then it removes the transmission schedule from this radio and provides the transmission schedule to the next one. Once the next ground-based radio receives a transmission schedule, it begins transmitting to the airborne TmNS radio.

Throughout this process, the airborne TmNS radio continue to use its transmission schedule in order to maintain its downlink flow of data. When the A2A handoff is conducted while two or more ground-based radios (e.g., the original radio and the next one that is being handed off to) are within range and able to receive the transmissions from the airborne radio, the A2A handoff event should be a lossless event. There may be a short delay in uplink traffic while the transmission schedule is moved from one radio to another, but the traffic transmitted from the TA networks to the ground is not directly affected with an A2A handoff.

Multiple copies of incoming traffic are received over the TSS tunnel connections of the LM during periods of time where multiple ground-based TmNS radios are receiving the transmissions from the airborne TmNS radio. The LM is responsible for detecting and filtering out any duplicate messages. The LM will only perform the IPv4 routing function on the first instance of a network packet it receives with one of its TSS tunnels; any duplicate network packets received are dropped. This duplicate packet detection and dropping function prevents the end network in the MCR from receiving the duplicated network traffic.

The LM determines the A2A handoff decision when automated handoffs are enabled. The LM may have several possible rules engines that can be used to determine when to perform the A2A handoff. Different rules can be selected for different test missions.

The LM maintains a table of automatic A2A handoff rules that the SysMgr can assign to a particular link destination group RF MAC address. The rule number is associated with a string representation of the rules engine to be used for the computation and decision, but the standard is silent on the format of this string representation or how to interpret it. The way that this capability has been demonstrated is for this string value to represent the name of an executable script or program to run. The inputs to the program are associated with the link statistics that are being evaluated for the handoff decision.

An example rule is one that checks the most recent received signal strength indication (RSSI) values for all ground-based TmNS radios receiving from a specific destination group RF MAC address. It assumes that the radio that has the highest RSSI value has the best view of the TA. The rule reviews the RSSI values for all associated radios every second. If a ground radio that is not currently selected for transmitting to the TA registers five consecutive RSSI values stronger than the currently selected ground-based radio, then the LM will automatically execute the A2A handoff from the initially transmitting ground-based radio to the ground-based radio that has the stronger RSSI values. The five-second history built into this rule engine prevents the LM from performing rapid A2A handoffs unnecessarily when the RSSI values are relatively the same and the stronger RSSI value alternates between the two radios.

Other rules can be written in a similar fashion utilizing the desired link properties. The specifics of the rules engines are left to the ranges.

4.3.2.3.2 Link Manager's Role in N2N Handoffs

Two LMs are required during N2N handoffs. Each LM manages the transmission schedules for their respective RAN frequencies. This type of handoff should be manually initiated by a SysMgr user because it has greater potential to affect other active test missions.

The initial LM establishes TSS tunnel connections to each of the ground-based TmNS radios associated with the test mission. Likewise, the second LM establishes TSS tunnel connections to the ground radios that it manages links for. These sets of ground-based radios do not overlap. In addition to these TSS tunnels, the initial LM will also initiate a TSS tunnel connection to the second LM prior to performing the N2N handoff. The initial LM remains in the data path between the MCR and the TA network regardless of whether or not the handoff has occurred.

When manually commanded to perform the N2N handoff by the SysMgr user, the initial LM requests a transmission schedule from the second LM to be used by the TA radio. Once received, the initial LM changes the uplink route toward the airborne network by sending through the TSS tunnel to the second LM. The initial LM then sends a schedule update message to the TA radio that does two things: 1) removes any transmission opportunities for the RAN 1 frequency from the radio's schedule; and 2) provides the transmission opportunity for the RAN 2 frequency obtained from the second LM to the TA radio. When the TA radio applies these transmission schedule changes, the TA will have been handed off to RAN 2 and will be transmitting on the new frequency. At this point, the second LM will be able to connect to the TA radio and begin providing a dynamic transmission schedule that fits within the current scheduling load of RAN 2.

After the N2N handoff, the ground-based radios of RAN 2 will route the incoming airborne traffic through their TSS tunnels to the second LM, which will in turn route the data through its TSS tunnel to the initial LM, which will then be able to route the traffic to the MCR. By keeping the initial LM in the data path, it prevents other devices from having to be aware of the route changes in the network that are associated with any handoff events.

4.3.2.3.3 Link Manager's Role in R2R Handoffs

Two LMs are required, one located in each range, during R2R handoffs. The frequency of the RANs used by the two LMs may or may not be the same. The general concept of the R2R handoff is very similar to the N2N handoff with the caveat that the initial LM may not need to request a transmission schedule from the second LM prior to performing the handoff. Due to the nature of R2R handoffs, the geographic separation of the test ranges involved may not overlap coverage areas. As such, the handoff responsibility from the initial LM is to clear the transmission schedule of RAN 1 from the TA radio. The SysMgr can then send a command to change the TA radio frequency to the RAN 2 frequency. Note when this happens, an SNMP timeout is expected to occur, as there will be no active transmission schedule for the SNMP response to be transmitted back. Even still, the command may have succeeded.

The initial LM will initiate a TSS tunnel connection to the second LM. After the handoff event occurs, the received network traffic will flow from the second LM on the second range through the TSS tunnel to the initial LM on the first range, and from there to the destination on the first range. Likewise, data destined for the TA network after the R2R handoff event has occurred will continue flow to the initial LM as the next hop router to the TA network. The initial LM will route the data through the TSS tunnel to the second LM en route to delivery to the TA network.

If the two ranges involved in the R2R handoff are geographically close and have overlapping coverage areas, it is possible to perform the R2R handoff in similar fashion to the N2N handoff with respect to obtaining an actual transmission schedule for the RAN 2 frequency.

4.3.3 Network Configuration and Time Synchronization

The TmNS radios require tight time synchronization in order to utilize their allocated transmission opportunities. This can be accomplished through either a direct GPS connection to each radio, or it can come by way of IEEE 1588-2008 (PTP) packets over the wired network segment. If utilizing PTP over the network, then it is crucial that the network switches in the path between the PTP grandmaster and the radio support the IEEE 1588-2008 protocol in order to provide a path for sub-microsecond synchronization on the radios.

In the examples provided, the TmNS radios used on the airborne TA synchronize their internal clocks directly to GPS by way of a dedicated GPS signal input into the radio. However, the TmNS radios used on the ground-based networks utilize IEEE 1588-2008 for time synchronization. They indirectly synchronize to GPS time through the PTP GMC on the local ground network segment, which is locked to a GPS signal. In this environment, the network path between the PTP GMC and the radio needs to be fully PTP-aware in order to support the required timing accuracies in synchronization needed.

For either fixed or mobile antenna sites, a typical ground-based installation is likely to contain the TmNS radio with a network switch and a PTP GMC. There may be other components on the network as well, including other switches. Connecting the TmNS radio and the PTP GMC device to the same network switch would only require one switch to support the IEEE 1588-2008 protocol. The PTP-aware switches can support the IEEE 1588-2008 protocol through either a boundary clock approach or a transparent clock approach.

Because IEEE 1588-2008 support is not a unique TmNS-specific capability, non-TmNS network switches can be used on the ground network as part of the transport path between the PTP GMC and the TmNS radio. However, the configuration of a non-TmNS switch is out of scope of the SysMgr and will need to be handled out-of-band.

Across a test range, it is likely that different antenna sites will have different IP spaces separated by routers. For each IP space, a different PTP GMC should be provided. Sites with multiple antennas and radios that are connected on the same network segment (e.g. within the same IP space) can use the same PTP GMC, but the same requirement for a fully PTP-aware network path between grandmaster and radio still applies.

4.3.4 System Manager

For the two-way telemetry capability, the SysMgr application provides a GUI for the user to be able to generate the MDL configuration files. For dynamically scheduled links, this includes radios and an LM. The GUI allows for the RF link associations to be made between radios. There is no need for a user to create static schedules for any of the links in the RAN as they will be provided by the associated LM. The LM application is responsible for ensuring that transmission assignments for the different links do not overlap within the epoch, that latency requirements can be met, and that bandwidth requirements are being met.

The SysMgr application may also be used during flight test operations to perform live health and status monitoring of the LM and the TmNS radios as well as performing some command and control operations of these components. For instance, a user of the SysMgr can initiate a handoff event through issuing some system management commands to the LM. It is not required to have a SysMgr application running in the network, but it is useful for situational awareness of the RAN, handoff control, and network troubleshooting.

This page intentionally left blank.

CHAPTER 5

Instrumentation Control Capability

A remote instrumentation control capability allows a user within an MCR to remotely control the operational status of TmNS components onboard the TA. Instrumentation control includes but is not limited to starting and stopping recorders, enabling and disabling the TDM delivery from DAUs, and reconfiguring components.

This capability builds off of a system that contains both a data acquisition capability (see Chapter 3) and a two-way telemetry link (see Chapter 4). With a communication path over the RF network established, the system management interfaces of the TmNS components onboard the TA are available for a SysMgr to monitor and control as a user chooses.

The TmNS components are controlled over the network by a SysMgr via SNMP. Other protocols may be available, such as the debug web interfaces of TmNS components over HTTP or HTTPS, but this chapter will focus on the management resources specified in Chapter 25.

This chapter will discuss the most common controls as well as the potential impacts to the system performance with exercising each of these controls. In each case, a SysMgr application is used to communicate with the TmNS instrumentation components.

5.1 Recorder Control

The most common remote control scenario is likely to involve changing the record mode of the onboard TA recorder. The operator can start and stop the recording function on the recorder from the SysMgr application within the MCR. When stopped, no data is written to the recorder's media. Only after the recorder is commanded to begin recording will data begin being stored on the media. Depending on the purpose of the test, this capability can allow the operators the ability, should they so choose, to only enable data recording prior to the specific events being tested. The current practice is to start recording on the recorder at the beginning of the flight and then stop it at the end of the flight, recording everything in between. There is value in this in case something unexpected happens between test points. However, as data rates continue to climb on test programs, the ability for the test engineer on the ground to remotely control when the recorder begins stops recording can save the record media for the data that matters most.

In addition to controlling the recording function of the recorder, the media can also be remotely erased. For flight testing, this is probably something that will not be done often. In fact, the SysMgr application will provide confirmation dialogs in order to ensure a deliberate request to erase the media on a recorder. One possible use case could be during pre-flight operations to clear the media following any test recording made during pre-flight.

The recorder contains other status variables that can be retrieved and displayed by the SysMgr to the user. This includes the current record mode, media status and remaining capacity, and TDM statistics.

5.2 DAU Control

There are only a couple of control variables specific to a DAU, and perhaps only one of those would be considered for use during a flight test mission. A DAU can be remotely commanded to generate the TDMs onto the network or to not generate them. If a particular DAU is not needed for a specific test, a SysMgr user could simply turn off the TDM output from the DAU, which will reduce the overall network chatter on the TA network.

Enabling and disabling the TDM output from DAUs may be beneficial during pre-flight checkout operations or when trying to isolate certain network problems. During a flight, if it becomes obvious that a DAU is generating errant data, the SysMgr user can opt to disable the TDM output if the absence of the data from this one DAU is not detrimental to the rest of the test. The decision to do so, of course, would be up to the test conductor.

The DAU contains other status variables that can be retrieved and displayed by the SysMgr to the user. This includes whether or not the DAU is in the mode to generate TDMs as well as statistic counters on the TDMs generated onto the network by the DAU. The statistic counters are specified by the MDID of the message and the multicast IP address it is transmitted to.

5.3 SST Transmitter Control

The TmNS SST transmitters can be controlled through the TMNS-MIB as well. These controls include enabling and disabling the RF transmissions as well as the transmitter's operating frequency.

Most, if not all, SST transmitters used today are not TmNS-compliant. In order to control them through the TmNS defined processes, an intermediate proxy device will need to be utilized. Initial test configurations had this capability built into the TmNS radios. These specific TmNS radios advertise through the TMNS-MIB that they support SST transmitter capabilities. For any commands received that are specific to the SST transmitter, the TmNS radio would supply the corresponding Chapter 2 command over a serial connection it maintains to the non-TmNS SST transmitter.

5.4 Device Reconfiguration Control

Configuration of TmNS components is done over the network according to the configuration protocol described in Chapter 25 Subsection 25.4.3.1. To sum up the protocol, it states that SNMP will be used to initiate the configuration process on the TmNS component. When initiated, the component being configured will retrieve its MDL configuration file from the network location specified through another TMNS-MIB variable.

Having a two-way link that provides network connectivity between the airborne TA network and the ground-based MCR network enables the capability of the SysMgr application to command any of the instrumentation components onboard the TA to reconfigure. However, just because the capability exists does not suggest that it should be utilized often. With the two-way link, this capability is now possible, but there are no specific concepts of operations (CONOPSs) yet requiring such a capability. Future CONOPSs may require this capability.

There are a couple scenarios that may serve as a good reason to reconfigure a TmNS instrumentation component. One case is that the current flight's testing requirements have been completed but there is an opportunity to keep flying. The instrumentation can be reconfigured for the next day's test mission configuration and be ready for use within a matter of minutes. The other case would be when a particular instrumentation component is misbehaving due to possible manual configuration changes, and it is necessary to reset the configuration to a known state. There may be other cases as well.

Due to the potential impacts of reconfiguring a component on the network, some best practice guidance is included in the following subsections.

5.4.1 Minimize Traffic Across RF Network

It is best to generate and store the MDL file on board the TA network such that each component to be reconfigured will not need to retrieve the file over the RF link. Doing this will be much faster, and after the initial file transfer, it will only require RF bandwidth usage for the SNMP commands from the SysMgr to set the MDL file location on each device and then the configuration command.

The recorder is the likely resource on the TA network to store a file. It should have a general file system partition that is separate from the main data recording partition. A TmNS recorder that is running a file transfer protocol (or secure file transfer protocol) server would be the ideal location for storing the MDL files for reconfiguration.

5.4.2 Impacts of Reconfiguring a DAU

When reconfiguring a DAU, one can expect it to reboot in the process. It should be well understood that the DAU will stop generating TDMs during the course of the reconfiguration and reboot. This should be acceptable in either case mentioned above because either the current test is complete and additional data is not needed, or the DAU is in a bad state and may be putting out bad data. In either case, the reboot event will cause the TDM counter statistics to be reset. This could result in the TmNS recorder reporting a bunch of lost TDMs in the process. The lost TDMs would be based on TDM sequence numbers received by the recorder, and the lost TDMs counter would trace it to the particular DAU that was rebooted to verify this fact.

5.4.3 Impacts of Reconfiguring a Recorder

When reconfiguring a recorder, one can expect the recorder to reboot in the process. Any active record sessions should be stopped by a user prior to reconfiguring. After the reconfiguration, all TDM counter statistics on the recorder are reset to zero. If the TDM counter statistics are important from the point before the reconfiguration, they should be retrieved prior to the reconfiguration as well. Because these TDM counters are reset to zero, it seems practical to reconfigure the recorder after all other TDM-generating instrumentation has been reconfigured, if indeed other components are being reconfigured.

5.4.4 Impacts of Reconfiguring a Radio

As a general rule, it is never a good idea to reconfigure an airborne TmNS radio. If this is done, the two-way link will obviously be lost, at least for some time until the radio reboots. Because the consequences can be severe, such as the link never being re-established, the rationale for a reconfiguration should be well thought-out. Can the change be made through other means via other specific management resources?

The MDL configuration files for radios can contain static schedules for their RF links. If a TA radio operating on a RAN with no LM available needed a scheduling change, it could be reconfigured with a high level of confidence that the link would be re-established after the reconfiguration if only the transmission schedule was changed. If this is the scenario, then it is likely that there are other radios that may be receiving scheduling changes as well at the same time. Caution must be taken to reconfigure radios in an order such that there is no time window in which two radios share the same transmission window.

Reconfiguring ground-based radios is expected to be a daily occurrence. It is expected that each ground radio will be reconfigured for each test mission that it will support, which may be multiple missions each day. Because connectivity to these ground-based assets is via the ground-based infrastructure, there is less risk of a bad configuration causing the component to become unreachable. Adding a new ground-based radio asset to a mission during the middle of the mission is also OK to do and may even become common practice.

Ground-based radios may be reconfigured at any time without any significant impact to the system unless the radio is actively being scheduled by the LM and is providing the uplink portion of the two-way link. If this is the case, then a reconfiguration of the radio will cause the link to drop out until the radio reboots and can re-establish the link. If another ground-based radio is available in this test mission and in position for an A2A handoff, a manual handoff command should be sent to the LM. After the handoff, the former ground-based radio can be reconfigured without consequence of disrupting the test mission.

5.4.5 Verify New Configuration Versions

Any time a TmNS component is reconfigured, the user should verify that the configuration attempt was successful and that the new configuration has been applied. This is easily observable with the SysMgr application that initiates the reconfiguration. As part of the device configuration protocol, the device should send an SNMP notification to the SysMgr application to inform the user that the configuration attempt has completed and whether or not the attempt was successful.

Alternately, there are management resources that can be retrieved on any TMA that specify the current configuration version, the time of the last configuration attempt, and the result of the last configuration attempt. These should provide confidence to the user that initiates the component reconfiguration that the components are up and running as expected.

Any configuration failures will need to be addressed immediately by the user to determine the problem and make the appropriate fix.

CHAPTER 6

Data Retrieval Capability

A TmNS system with a data retrieval capability builds off of a system that contains both a data acquisition capability (see the Data Acquisition Capability in Chapter 3) and a two-way telemetry capability (see the Two-Way Telemetry Capability in Chapter 4). Together, these two capabilities provide an instrumentation system onboard the TA that is network-accessible from the MCR on the ground. A data retrieval capability can be realized with a TmNS recorder that provides a data retrieval interface and a data retrieval client by which the requests can be made.

This chapter will describe the different ways that data can be requested for retrieval. It will cover the TmNS components directly involved with data retrieval along with any specific requirements for this capability. It will also include a discussion of specific CONOPS that make use of the data retrieval capability, namely the "PCM Backfill" scenario for automatically retrieving data due to PCM dropouts and the "Fetch Data" scenario requesting data not initially being telemetered.

6.1 Granularity of Data Retrieval Requests

The TmNS data retrieval protocols are defined in Chapter 26. Data retrieval requests include the data to be retrieved and a valid time range associated with the data being requested. The different options are discussed in further detail in the following subsections.

6.1.1 Time-Based Requests

For each data request, a start time is provided and may include an optional stop time. These serve as the time range by which the returned TDMs contain data within the range.

A start time may consist of a specific system timestamp, or it may utilize one of two labels: the "start" label to indicate the range begins with the earliest message timestamp of all available TDMs; or the "now" label to indicate the time range begins at the receipt of the data retrieval request.

The stop time may consist of a specific system timestamp, or it may utilize either the "now" label or "end" label, both of which return the same result and end the request with the timestamp at which the data retrieval request was received.

An open-ended request is one in which no stop time is specified. An open-ended request allows for the retrieval channel to remain open and to put future acquired data that matches the request criteria into the channel until the channel is closed.

On a similar note, a stop time may be provided as a timestamp in the future. In this case, the channel will remain open and deliver any data arriving in the future that matches the request criteria until the stop time is encountered.

Different start time and stop time combinations are used in support of different CONOPS, such as PCM-backfill or mid-test decision to begin monitoring data that was not initially configured as part of the telemetry stream.

6.1.2 Message-Level Requests

The basic structure of a TDM has already been discussed (see Section 3.1). Simply put, a message contains packages, and packages contain measurements. Data retrieval requests can be made at any of these three levels: message (MDID), package (PDID), or measurement (MeasID).

The MDID-based data retrieval requests are the simplest. Requesting an MDID is a one-for-one transaction. For each message with the corresponding MDID, the whole message is retrieved and sent to the requestor.

Depending on the content of the MDID being requested, an MDID-based request may not be the most network-efficient means of retrieval. This would be the case if an MDID contains multiple PDIDs but only a single PDID is of interest for retrieval.

6.1.3 Package-Level Requests

For a PDID-based data retrieval request, only the requested PDIDs are to be returned to the data retrieval client. However, PDIDs must be transported within a message, so the request will indicate the MDID over which the requested PDIDs will be delivered by. The delivery MDID may be a specific MDID used only for data retrieval purposes. It may use its original MDID only if the description of the MDID can be satisfied with only the requested PDIDs.

Utilizing PDID-based data retrieval allows for a more granular set of data than what is afforded by MDID-based requests. This can lead to an improved network performance over MDID-based requests, but it incurs a little more complexity in the system. For instance, new TDM definitions (with unique MDIDs) may need to be constructed in order to be able to take advantage of the network efficiencies.

It should be noted that not all TmNS recorders that support some level of data extraction will support this type of request. It is not guaranteed that a TmNS recorder that supports MDID-based requests will support the PDID-based requests. Consult the vendor documentation for specific data retrieval capabilities of the TmNS recorder.

6.1.4 Measurement-Level Requests

Measurement-level retrieval requests, also referred to as MeasID-based requests or parametric data extraction, is the most granular form of TmNS data retrieval. As its name suggests, this level of data retrieval is specific to extracting and retrieving specific measurements from packages. For delivery, the measurements are placed into a delivery PDID that is then in turn placed into a delivery MDID for the package. The delivery PDID and delivery MDID may be defined specifically for data retrieval purposes.

Network efficiencies can be gained through parametric data extraction by only delivering the exact measurements of interest rather than including other measurements contained in the original package but are not of interest to the requesting software. While parametric data extraction can prove to be the most network-efficient means of retrieving data from the onboard recorder, it is also the most complex, particularly for the recorder in terms of searching and finding the requested data.

It should be noted that not all TmNS recorders that support some level of data extraction will support this type of request. It is not guaranteed that a TmNS recorder that supports PDID-based requests will support the MeasID-based requests. Consult the vendor documentation for specific data retrieval capabilities of the TmNS recorder.

6.2 TmNS Components and Requirements for Data Retrieval

Retrieving data from the system assumes that there is at least one TmNS DAU producing data that is being consumed by a TmNS recorder. This section only focuses on the TmNS components directly involved in the data retrieval process. They include the following.

- TmNS Recorder
- Data Retrieval Client

The previously mentioned example MDL file, here, contains a TmNS recorder that can provide data retrieval services upon request.

For ground-based testing or lab testing, data retrieval can be accomplished over a direct Ethernet connection to the TmNS radio rather than having to utilize the two-way telemetry link. The retrieval capability really resides in the TmNS radio and the data retrieval client; the two-way telemetry link is simply just the transport that supports the data delivery.

6.2.1 TmNS Recorder

The TmNS recorder needs to support the RC data source capability in order to provide a data retrieval capability. A recorder can only provide data that it has recorded, so it is of foremost importance that the recorder is configured to record the data that it may be expected to reproduce upon receiving a data retrieval request. See Chapter 3 for information concerning recording TDMs from DAUs with recorders.

Recorders may have different capabilities when it comes to performing data retrieval. Any TmNS recorder with RC data source capability should be able to service the simplistic MDID-based requests; however, PDID-based and MeasID-based requests become more of a challenge for different recorders. The capability of the particular recorder should be known prior to defining the messages, packages, and measurements made available for servicing data requests. The burden of the complexity of these requests falls largely on the recorder as it may potentially need to create new packages and then new messages from those packages to service some data requests rather than just sending over pre-recorded whole messages at the message level. Any new messages or packages that are to be used exclusively for data retrieval must be defined in the MDL configuration file of the recorder.

The recorder responds to RC data requests by producing the requested TDMs to the requesting data client. Because the type of request, whether MDID, PDID, or MeasID, is most relevant to the data retrieval client making the request, it will be discussed in the next section describing the data retrieval client.

6.2.2 Data Retrieval Client

A data retrieval client is a piece of software that generates data retrieval requests via the RC data request channel as described by Chapter 26. This software can be a stand-alone application that performs the data retrieval requests of a recorder, or it can be functionality embedded within other software.

There are several different use cases for performing data retrieval. One use case is to provide a PCM backfill capability, which is described separately in Section 6.3. Another use case is to perform an open-ended data retrieval for data that was not being telemetered at the start of the test. This scenario is described in Section 6.4. These use cases are examples of how the data retrieval capability can be utilized over the two-way telemetry link.

6.3 PCM Backfill

During flight testing, it is inevitable that the PCM stream will at some point drop out, particularly during the time when the TA is performing certain maneuvers. During these times the data collected is of most interest. The PCM backfill operation is one approach to address this scenario. When a PCM dropout event is detected on the SST data stream, two things are known: the specific measurements that experienced some data loss; and the time window during which the lost messages exist. Based on this information, the PCM backfill operation can be automatically carried out over the TmNS link through an RC data request for the specific measurements during the specific time window over which the dropout occurred.

The PCM backfill scenario is one example of a CONOPS that utilizes the TmNS data retrieval protocols to fill in missing PCM data. Automating these requests is an efficient manner of retrieving only the data that is needed on the ground without having to involve a user to generate the request manually. This capability has been demonstrated with a data retrieval client embedded alongside a digital strip chart display software. Figure 6-1 shows a strip chart display application that is utilizing a PCM backfill capability to retrieve data dropouts from the PCM stream.

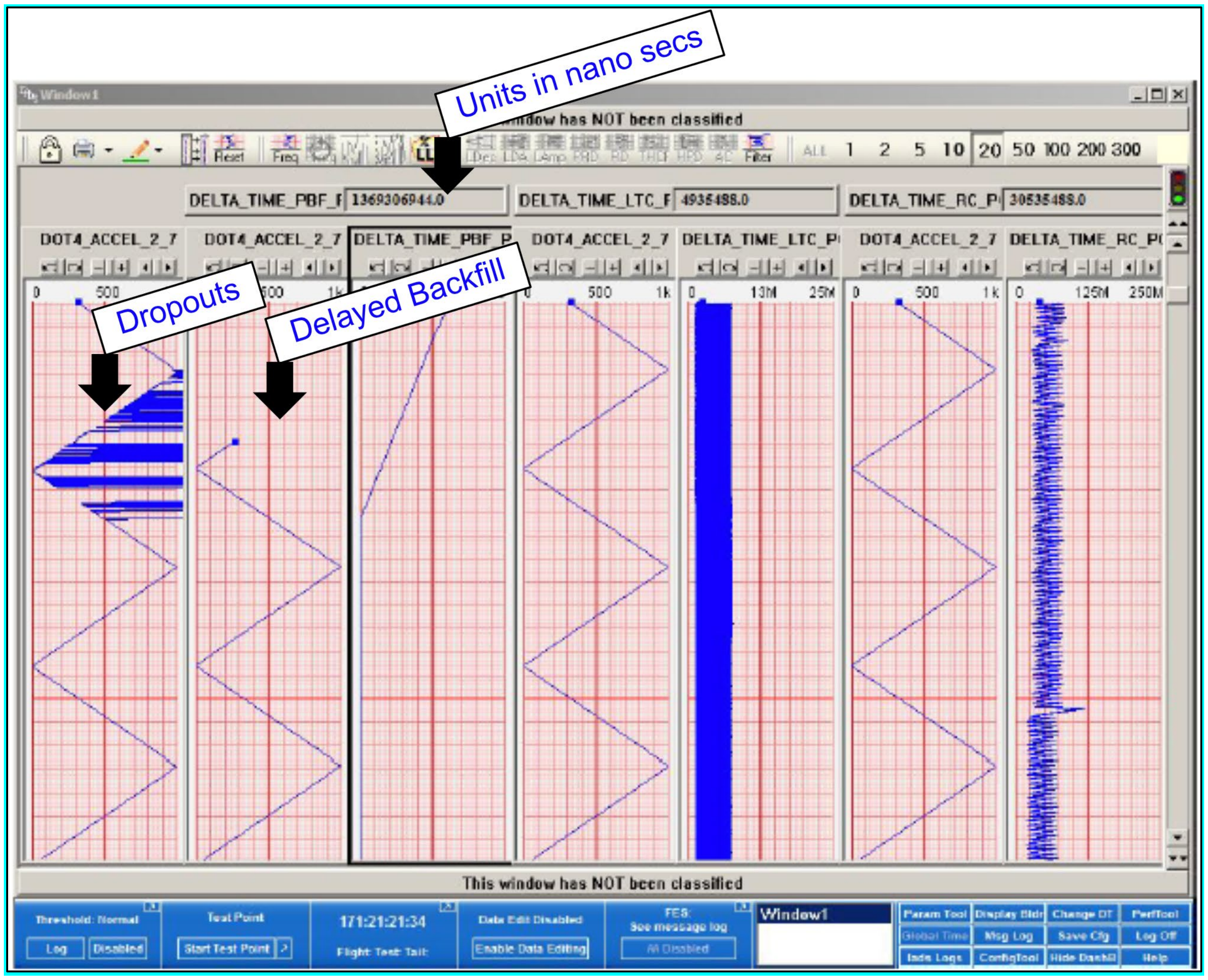


Figure 6-1. Strip Chart Demonstration of PCM Backfill in Progress

6.4 Fetch Data

The fetch data scenario utilizes the TmNS link to deliver data that was not originally defined to be telemetered during a flight test. The PCM data streams are defined at configuration time, which is prior to takeoff. However, during the course of a test flight, if an additional measurement is desired to be retrieved from the TA, it can be retrieved over the TmNS link. A data retrieval client can submit an open-ended request for a specific measurement (or package or message) from the onboard TmNS recorder, and the recorder will send the appropriate messages over the TmNS link as they are received and recorded.

Though the fetch data scenario utilizes the same protocols for retrieving data as the PCM backfill scenario, this scenario requires the user to identify data and then initiate the retrieval of that data whereas the PCM backfill does not require any user interaction for retrieving the dropped-out data.

This page intentionally left blank.

CHAPTER 7

Voice over IP Capability

The VoIP capability can be used over the two-way TmNS link in order to provide voice communications between the TA and the ground. In order to have a VoIP capability, a two-way link is required (see Chapter 4).

The system will also need to incorporate VoIP components on the airborne platform as well as in the ground network, such as within the MCR. The VoIP components can be off-the-shelf components, or they may be TmNS-compliant devices, such as a TmNS voice gateway component.

7.1 Components Required for VoIP Capability

Calls over VoIP consist of two or more participants. Each participant connects to a Session Initiation Protocol (SIP) server for establishing or joining a VoIP conversation. The protocols used for VoIP are not specific to the TmNS; thus there may be several off-the-shelf components that can supply this capability. A TmNS component would include an MDL configuration interface as well as a TmNS manageable application management interface.

The MDL configuration file for a TmNS VoIP gateway component contains phone numbers, supported codecs, and the SIP server IP address.

If non-TmNS components are used for this capability, configuration of those components is performed out-of-band.

7.1.1 TmNS VoIP Gateways/SIP Clients

A TmNS VoIP gateway component operates as a SIP client that connects to a SIP server for establishing and maintaining VoIP calls over the network. In addition to the SIP client capability, the TmNS VoIP gateway is manageable via the TMNS-MIB and is configurable via an MDL configuration file. The SIP client capability is the minimum capability required for VoIP call establishment.

Many COTS products, both hardware and software, are available to provide a standard SIP client capability. If a COTS product is used in the system to provide the VoIP capability, the configuration and management of the product are out-of-band and not within the scope of the TmNS. In these cases, this component will require manual configuration through the vendor-provided mechanisms.

7.1.2 SIP Server

The SIP server is the centralized call manager for a VoIP system. The SIP client endpoints register with the server and can initiate calls through the server. When a call is initiated, the server rings the remote client that is being called. The server can conference in multiple participants into a single call. The server can also manage multiple calls at one time.

A SIP server may be a dedicated hardware component, or it may be a piece of software running on a computer. Many hardware components with SIP server functionality often have local SIP clients as well.

In most cases, the SIP server should be located on the ground network rather than on a TA. This choice will most often lead to a better performance over the RF network, especially if there are multiple ground participants. It also performs better for VoIP calls that consist of two or more TA participants in a non-relay scenario. In a relay scenario with both aircraft operating as VoIP call participants, better RF network performance for VoIP-related traffic may be achieved through placing the SIP server capability onboard the intermediate relaying aircraft.

7.2 Performance Characteristics and Requirements

Performance of VoIP through the system is largely summed up through minimizing the network latency through the RF network. Codec selection also plays a role.

7.2.1 Network Latency

Minimizing the end-to-end latency of VoIP traffic is crucial in order to support a VoIP-based conversation. Too much delay results in poor conversation between participants. While end-to-end delay would be most accurately measured from the time one participant speak to the time the other participant(s) hear, it should be noted that delay can occur at numerous stages throughout the transport. The VoIP network packets can be delayed at each switch or router along the path as well. There are two things that the TmNS provides as a means of managing the delay across the network: QoS markings in the packets, and RF network scheduling characteristics.

7.2.1.1 DSCP

Voice traffic requires a low latency for delivery. Low latency traffic is often assigned to the DSCP category with an Expedited Forwarding per-hop behavior. This category is the highest data traffic category and provides low loss, latency, and jitter. When configured with an appropriate QoS policy, the TmNS radio will prioritize voice traffic to send ahead of other lower-priority traffic.

The TmNS identifies the DSCP value of 40 for voice traffic, which corresponds to a Class 5 class of service. Only VoIP traffic should use this DSCP marking in the system. The recommended DSCP traffic classifications for TmNS-based systems are provided in Appendix 22-A.

7.2.1.2 RF Network Scheduling for Low Latency

Due to the TDMA nature of the RF network, TmNS radios will buffer data that needs to be transmitted over the RF network whenever they are not transmitting. This buffering time translates into network latency. Assuming VoIP traffic is marked with the appropriate DSCP marking and the TmNS radios are configured with a QoS policy that prioritizes the VoIP traffic based on that DSCP marking, then the worst-case one-way network latency through a specific radio is approximately the duration of its largest time spacing between two TxOps. This is represented by the largest time gap between the end time of one TxOp and the start time of the next TxOp for that particular link.

Regardless of static or dynamic scheduled links, the transmission schedules need to be scheduled such that the one-way latency of VoIP packets across the network are minimized, generally kept less than 50 ms over the TmNS RF link. In order to achieve this, the resulting transmission schedules may contain more TxOps with shorter durations. While there are some network inefficiencies associated with smaller TxOps more frequently, they are needed in order to lower the worst-case latency over the RF link.

If manually building a static schedule, the user must ensure that TxOps are provided often enough for a link in order to maintain an acceptable VoIP latency. If the system relies on the LM to provide a dynamic schedule, the LM will be responsible for generating schedules that can satisfy the latency constraints specified in the service-level profiles of the QoS policies that each link it manages uses.

7.2.2 Codecs

The TmNS identifies the following codecs for use with VoIP systems.

- ITU-T G.711 – Pulse Code Modulation (PCM)[11]
- ITU-T G.726 – Adaptive Differential Pulse Code Modulation (ADPCM)[12]

In order to use a specific codec, both VoIP devices must support the codec.

[11] International Telecommunication Union. "Pulse Code Modulation (PCM) of Voice Frequencies." ITU-T G.711. 25 November 1988. May be superseded by update. Retrieved 20 January 2022. Available at https://www.itu.int/rec/T-REC-G.711/en.

[12] International Telecommunication Union. "40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)." ITU-T G.726. 14 December 1990. May be superseded by update. Retrieved 7 July 2020. Available at https://www.itu.int/rec/T-REC-G.726/en.

This page intentionally left blank.

CHAPTER 8

TA-to-TA Relay Capability

A TA-to-TA relay capability can be used to provide a two-way over-the-horizon TmNS link between a TA and the ground. A TA-to-TA relay capability requires a two-way link (see Chapter 4).

This chapter provides guidance on link configuration for the relay scenario. It also provides network performance implications of conducting relay operations.

In the example network topology, RAN1 includes a test mission that exercises the TA-to-TA relay capability. The radio transmission schedules in this example are dynamically provided via an LM. The example MDL configuration file can be found here.

Figure 8-1 shows the portion of the example topology used for the TA-to-TA relay scenario. It includes one ground-based TmNS radio and two airborne TmNS radios.

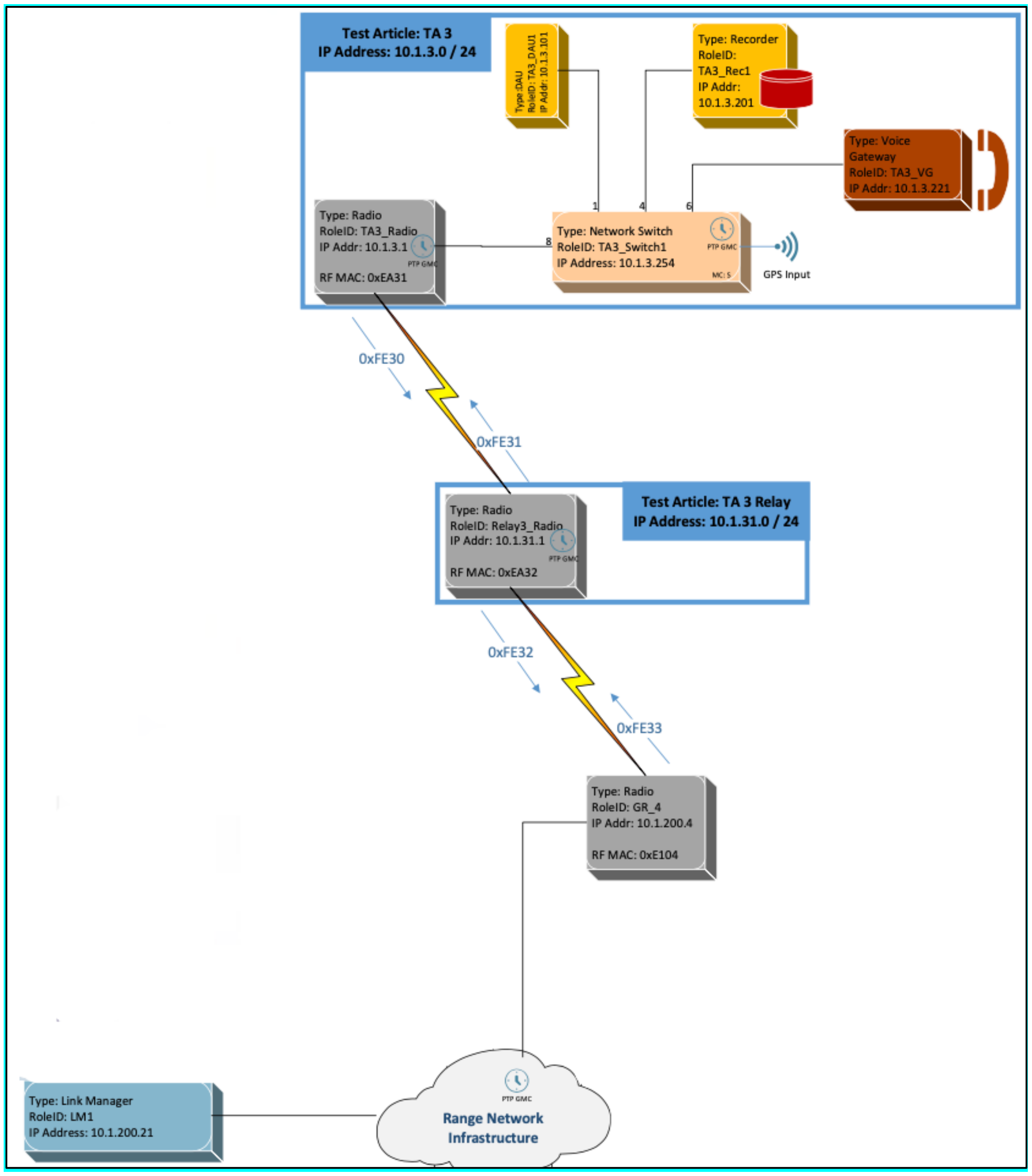


Figure 8-1. Example Topology for a Dynamically Scheduled TA-to-TA Relay Capability

8.1 TmNS Components for TA-to-TA Relay Capability

In order to extend the two-way telemetry capability described in Chapter 4 into an airborne relaying capability, one need only to supply a second TA to perform the RF network relay. This second TA can be as simple as containing only a TmNS radio (with time synchronization, of course) to be used only for relaying for the instrumented TA, or it can be an instrumented aircraft as well. In the example provided in this handbook, the airborne relay system consists only of a TmNS radio that is time synchronized to GPS.

8.2 RF Link Configuration

Relay capability can be provided with either static or dynamic link scheduling. The key to configuring the RF links properly is to ensure each TmNS radio is subscribed to the transmissions from the radios that they are expected to receive transmissions from. For the relaying radio, this may include subscribing to two links, one from the ground-based radio and one from the TA-based radio. The relaying radio will also utilize multiple transmission links, one for transmissions to the ground-based radio and a different one for transmissions to the TA-based radio.

The previously mentioned example MDL file, here, contains TmNS radios supporting a TA-to-TA relay scenario.

8.3 Performance Considerations

The relay capability provides the ability to perform two-way telemetry over the horizon; however, there are performance costs associated with doing this. There are direct impacts to end-to-end throughput as well as latency for the radios utilizing a relayed approach.

8.3.1 Throughput Performance Considerations Through Relay

When performing a relay service over the RF network, it should be understood that sending data from the ground network to the TA network via the relay will require two RF network transmissions: one from the ground-based radio received by the relaying radio; and the other from the relaying radio to the TA radio. A similar requirement exists for transmissions from the TA network to the ground network via the relay. On its surface, this requires twice the bandwidth needed compared to the non-relayed approach.

8.3.2 Latency Performance Considerations Through Relay

In order to meet end-to-end latency requirements over the RF network through a relay, the transport time includes buffering at both radios that transmit the data across the network. The transmission schedule needs to provide opportunity for the relaying radio to transmit its received data to the destination radio in time to meet the latency requirements.

This page intentionally left blank.

CHAPTER 9

Checkout and Troubleshooting TmNS-Based Systems

NOTE	This section is under development. The following text indicates what is planned for inclusion in a future update.

Executing basic checkout procedures during system build-up and before test missions can ensure that the system is functional prior to the test mission. Any issues found can be resolved prior to the test mission execution. The verification steps give system engineers confidence that the system is performing as expected.

Some basic tools that are worth having available include the following.

- Wireshark
- MIB browser
- Oscilloscope for timing verification

Checkout procedures may include the following.

- Verifying TDMs with Wireshark (measurement-specific details with MDL-based Wireshark decoder)
- System Management verification and status
- Time synchronization status

When some portion of the system experiences a failure or exhibits unexpected behavior or performance, troubleshooting those issues is required in order to find the root cause of the issue.

NOTE: In a full wiki environment, this chapter will serve as a lessons learned section for users to update with their real-world experience in system checkout procedures and troubleshooting steps.

This page intentionally left blank.

APPENDIX A

Citations

Institute of Electrical and Electronics Engineers. *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*. IEEE 1588-2008. 24 July 2008. Superseded by IEEE 1588-2019. Retrieved 20 January 2022. Available at https://standards.ieee.org/standard/1588-2008.html.

International Telecommunication Union. "40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)." ITU-T G.726. 14 December 1990. May be superseded by update. Retrieved 7 July 2020. Available at https://www.itu.int/rec/T-REC-G.726/en.

———. "Pulse Code Modulation (PCM) of Voice Frequencies." ITU-T G.711. 25 November 1988. May be superseded by update. Retrieved 20 January 2022. Available at https://www.itu.int/rec/T-REC-G.711/en.

Range Commanders Council. "Management Resources." In *Telemetry Standards*. RCC 106-20 Chapter 25. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. "Message Formats." In *Telemetry Standards*. RCC 106-20 Chapter 24. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. "Metadata Configuration." In *Telemetry Standards*. RCC 106-20 Chapter 23. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. "Network-Based Protocol Suite." In *Telemetry Standards*. RCC 106-20 Chapter 22. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. "Radio Frequency Network Access Layer." In *Telemetry Standards*. RCC 106-20 Chapter 27. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. "Radio Frequency Network Management." In *Telemetry Standards*. RCC 106-20 Chapter 28. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. "Telemetry Network Standard Introduction." In *Telemetry Standards*. RCC 106-20 Chapter 21. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. “TmNSDataMessage Transfer Protocol.” In *Telemetry Standards*. RCC 106-20 Chapter 26. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

———. “Transmitter and Receiver Systems.” In *Telemetry Standards*. RCC 106-20 Chapter 2. May be superseded by update. Retrieved 20 January 2022. Available at https://www.trmc.osd.mil/wiki/x/EoPOBg.

APPENDIX B

Networking Concepts

This appendix provides reference material for users that need more information regarding standard networking concepts, protocols, and technologies.

B.1 Basic networking concepts

Internet Engineering Task Force. “File Transfer Protocol (FTP).” RFC 959. Updated by RFC 3659, RFC 7151, RFC 2640, RFC 2773, RFC 2228, and RFC 5797. October 1985. Retrieved 7 July 2020. Available at https://datatracker.ietf.org/doc/rfc959/.

———. “Internet Protocol.” RFC 791. Updated by RFC 2474, RFC 6864, and RFC 1349. September 1981. Retrieved 7 July 2020. Available at https://datatracker.ietf.org/doc/rfc791/.

———. “Transmission Control Protocol.” RFC 793. Updated by RFC 6093, RFC 3168, RFC 1122, and RFC 6528. September 1981. Retrieved 6 July 2020. Available at https://datatracker.ietf.org/doc/rfc793/.

———. “User Datagram Protocol.” RFC 768. 28 August 1980. May be superseded or amended by update. Retrieved 6 July 2020. Available at https://datatracker.ietf.org/doc/rfc768/.

Quality of Service (QoS)

DiffServ Code Points (DSCP)

Unicast, Broadcast, and Multicast

B.2 Network management protocols

Internet Engineering Task Force. “Simple Network Management Protocol.” RFC 1157. May 1990. Retrieved 24 January 2022. Available at https://datatracker.ietf.org/doc/rfc1157/.

B.3 Time synchronization protocols

Internet Engineering Task Force. “Network Time Protocol Version 4: Protocol and Algorithms Specification.” RFC 5905. June 2010. Updated by RFC 7822 and RFC 8573. Retrieved 28 July 2021. Available at https://datatracker.ietf.org/doc/rfc5905/.

*** * * END OF DOCUMENT * * ***