COORDINATED HIGHWAYS ACTION RESPONSE TEAM
STATE HIGHWAY ADMINISTRATION

CHART Release 8

Detailed Design

Revision 2

Contract SHA-06-CHART
Document # WO23-DS-001
Work Order 23, Deliverable 4

August 1, 2011
By
CSC

Revision	Description	Pages Affected	Date
0	Initial Release	All	5/23/2011
1	Changes based on comments made from MdTA review	4-9, 5-9	6/02/2011
2	Changes based on results of HAR field testing.	2-24, 3-1, 4-4, 4-9, 4-15, 5-4, 5-8, 5-9, 5-10, 5-13, 8-149, 8-155, 8-159	8/01/2011

Table of Contents

Table of Figures

1 Introduction

1.1 Purpose

This document describes the design of the software for CHART Release 8. This build provides the following new features:

- IP HAR: CHART R8 will add support for TCP/IP communications with the HIS DR1500 HAR. In addition to faster communications, use of TCP/IP allows for more detailed status information to be collected from the HAR and allows CHART to automatically poll the HAR and detect hardware failures. Device failure alerts and notifications will be added for HAR devices, although the system will only be capable of detecting hardware failures for the HIS DR1500, and only when it is controlled via TCP/IP. (The system will be capable of detecting communication failures regardless of HAR model and communication medium.)
- IP SHAZAM: CHART R8 will add support for a new SHAZAM device which communicates via TCP/IP. The existing automatic refresh feature that exists for the Viking SHAZAM device will be available for this new SHAZAM model and will be enhanced to detect hardware failures. Device failure alerts and notifications will be added for SHAZAM devices, although the system will only be capable of detecting hardware failures for the new SHAZAM model, the HWG-ER02a. The system will be able to detect communication failures for either SHAZAM model.

1.2 Objectives

The main objective of this detailed design document is to provide software developers with a framework in which to implement the requirements identified in the CHART R8 Requirements document. A matrix mapping requirements to the design is presented in Section 15 (Mapping to Requirements).

1.3 Scope

This design is limited to Release 8 of the CHART system. It addresses both the design of the server components of CHART and the Graphical User Interface (GUI) components of CHART to support the new features being added. This design does not include designs for components implemented in earlier releases of the CHART system.

1.4 Design Process

The design was created by capturing the requirements of the system in UML Use Case diagrams. Class diagrams were generated showing the high level objects that address the Use Cases. Sequence diagrams were generated to show how each piece of major functionality will be achieved. This process was iterative in nature – the creation of sequence diagrams sometimes caused re-engineering of the class diagrams, and vice versa.

1.5 Design Tools

The work products contained within this design will be extracted from the Tau Unified Modeling Language (UML) Suite design tool. Within this tool, the design will be contained in the CHART project, Release 8, Analysis phase and System Design phase.

1.6 Work Products

The final CHART Release 8 design consists of the following work products:

- Use Case diagrams that capture the requirements of the system
- Human-Machine Interface section which provides descriptions of the screens that are changing or being added in order to allow the user to perform the described uses.
- UML Class diagrams, showing the software objects which allow the system to accommodate the uses of the system described in the Use Case diagrams
- UML Sequence diagrams showing how the classes interact to accomplish major functions of the system
- Requirement Verification Traceability Matrix that shows how this design meets the documented requirements for this feature

This document incorporates both the IP HAR and IP SHAZAM features by providing a single Use Cases section followed by a Detailed Design section for each feature. The use case diagrams are in Section 4, followed by the Detailed Design (including Human-Machine Interface, Class Diagrams, and Sequence Diagrams) for the IP HAR feature in Section 5, and finally the Detailed Design for IP SHAZAM in Section 6.

2 Architecture

The sections below discuss specific elements of the architecture and software components that are created, changed, or used in CHART Release 8.

2.1 Network/Hardware

CHART Release 8 features do not impact the network or hardware architecture of the CHART System.

2.2 Software

CHART uses the Common Object Request Broker Architecture (CORBA) as the base architecture, with custom built software objects made available on the network allowing their data to be accessed via well defined CORBA interfaces. Communications to remote devices use the Field Management Server (FMS) architecture. Newer external interfaces such as the User Management web service, Data Exporter, and GIS service employ a web services architecture combining an HTTP request/response structure to pass XML messages.

Except where noted in the subsections below, CHART Release 8 features do not impact the software architecture of the CHART System.

- For CHART R8 the field communications to HAR and SHAZAM devices will include the ability to use a TCP/IP network connection to connect to devices which support this communication medium. The existing Telephony communication medium for devices which require it will remain in place.

2.2.1 COTS Products

2.2.1.1 CHART

CHART uses numerous COTS products for both run-time and development. There is one new COTS product being added in Release 8:

Product Name	Description
Tritonus version 0.3.6 (Open source implmentation of the Java Sound API)	CHART uses this open source library to perform audio conversions to make HAR audio that exists in the CHART system compatible with the HIS DR1500 IP HAR.

The following table contains existing COTS products that have not changed for CHART Release 8:

Product Name	Description
Apache ActiveMQ	CHART uses this to connect to RITIS JMS queues
Apache Jakarta Ant	CHART uses Apache Jakarta Ant 1.6.5 to build CHART applications and deployment jars.
Apache Tomcat	CHART uses Apache Tomcat 6.0.29 as the GUI web server.
Apache XML-RPC	CHART uses the apache xmlrpc java library 3.1.2 protocol that uses XML over HTTP to implement remote procedure calls. The video Flash streaming "red button" ("kill switch") API uses XML over HTTP remote procedure calls.
Attention! CC	CHART uses Attention! CC Version 2.1 to provide notification services.
Attention! CC API	CHART uses Attention! CC API Version 2.1 to interface with Attention! CC.
Attention! NS	CHART uses Attention! NS Version 7.0 to provide notification services.
Bison/Flex	CHART uses Bison and Flex as part of the process of compiling binary macro files used for performing camera menu operations on Vicon Surveyor VFT cameras.
bsn.autosuggest	The EORS integration feature uses version 2.1.3 of the bsn.autosuggest JavaScript code from brandspankingnew.net. This tool is freely available and is included as source code in the CHART GUI. It provides a simple JavaScript tool that can be associated with a text entry field. When the user types characters in the field, the tool waits until there has been no typing for a configurable number of milliseconds (to make sure the user is done typing) then places an AJAX call to a web server which can return suggested results that match the user entered text. The bsn.autosuggest tool then parses the results (XML or JSON) and displays a UI element that shows the user the suggestions and lets them select one of them by clicking on it. If a suggested element is selected by the user, a configurable JS method is invoked to allow the application to use the selected suggestion.
CoreTec Decoder Control	CHART uses a CoreTec supplied decoder control API for commanding CoreTec decoders.
Dialogic API	CHART uses the Dialogic API for sending and receiving Dual Tone Multi Frequency (DTMF) tones for HAR communications.
ESRI's ArcGIS Sever	CHART uses version 9.3 to serve maps over the Internet.
ESRI's MapObjects	CHART uses the Map Objects 2.4 for spatial algorithms.
Flex2 SDK	The CHART GUI will use the Flex2 SDK, version 3.1 to provide the Flex compiler, the standard Flex libraries, and

Product Name	Description
	examples for building Flex applications.
GIF89 Encoder	Utility classes that can create .gif files with optional animation. This utility is used for the creation of DMS True Display windows.
JAXB	CHART uses the jaxb java library to automate the tedious task of hand-coding field-by-field XML translation and validation for exported data.
JDOM	CHART uses JDOM b7 (beta-7) dated 2001-07-07. JDOM provides a way to represent an XML document for easy and efficient reading, manipulation, and writing.
JacORB	CHART uses a compiled, patched version of JacORB 2.2.4. The JacORB source code, including the patched code, is kept in the CHART source repository.
Java Run-Time (JRE)	CHART uses 1.6.0_21
JavaService	CHART uses JavaService to install the server side Java software components as Windows services.
JAXEN	CHART uses JAXEN 1.0-beta-8 dated 2002-01-09. The Jaxen project is a Java XPath Engine. Jaxen is a universal object model walker, capable of evaluating XPath expressions across multiple models.
JoeSNMP	CHART uses JoeSNMP version 0.2.6 dated 2001-11-11. JoeSNMP is a Java based implementation of the SNMP protocol. CHART uses for commanding iMPath MPEG-2 decoders and for communications with NTCIP DMSs, and Cameras.
JSON-simple	CHART uses the JSON-simple java library to encode/decode strings that use JSON (JavaScript Object Notation).
JTS	CHART uses the Java Topology Suite (JTS) version 1.8.0 for geographical utility classes.
Log4J	CHART uses the log4J version 1.2.15 for logging purposes.
NSIS	CHART uses the Nullsoft Scriptable Installation System (NSIS), version 2.20, as the server side installation package.
Nuance Text To Speech	For text-to-speech (TTS) conversion CHART uses a TTS engine that integrates with Microsoft Speech Application Programming Interface (MSSAPI), version 5.1. CHART uses Nuance Vocalizer 4.0 with Nuance SAPI 5.1 Integration for Nuance Vocalizer 4.0.
OpenLayers	The Integrated Map feature uses the Open Layers

Product Name	Description
	JavaScript API 2.8 (http://openlayers.org/) in order to render interactive maps within a web application without relying on vendor specific software. Open Layers is an open source product released under a BSD style license which can be found at (http://svn.openlayers.org/trunk/openlayers/license.txt).
Oracle	CHART uses Oracle 10.1.0.5 as its database and uses the Oracle 10G JDBC libraries (ojdbc1.4.jar) for all database transactions.
O'Reilly Servlet	Provides classes that allow the CHART GUI to handle file uploads via multi-part form submission.
Prototype Javascript Library	The CHART GUI uses the Prototype JavaScript library, version 1.6.0.3, a cross-browser compatible JavaScript library provides many features (including easy Ajax support).
SAXPath	CHART uses SAXPath 1.0-beta-6 dated 2001-09-27. SAXPath is an event-based API for XPath parsers, that is, for parsers which parse XPath expressions.
SQLServer JDBC Driver	CHART uses this driver to lookup GIS related data and also to store Location Aliases in SQL Server databases.
Velocity Template Engine	Provides classes that CHART GUI uses in order to create dynamic web pages using velocity templates, CHART uses Velocity version 1.6.1 and tools version 1.4.
Vicon V1500 API	CHART uses a Vicon supplied API for commanding the ViconV1500 CPU to switch video on the Vicon V1500 switch

2.2.1.2 Mapping

There are no Mapping Application related changes included in this release, and therefore no changes to COTS products for that application.

2.2.2 Deployment /Interface Compatibility

2.2.2.1 CHART

2.2.2.1.1 External Interfaces

This section describes the external interfaces in Release 8 of the CHART system.

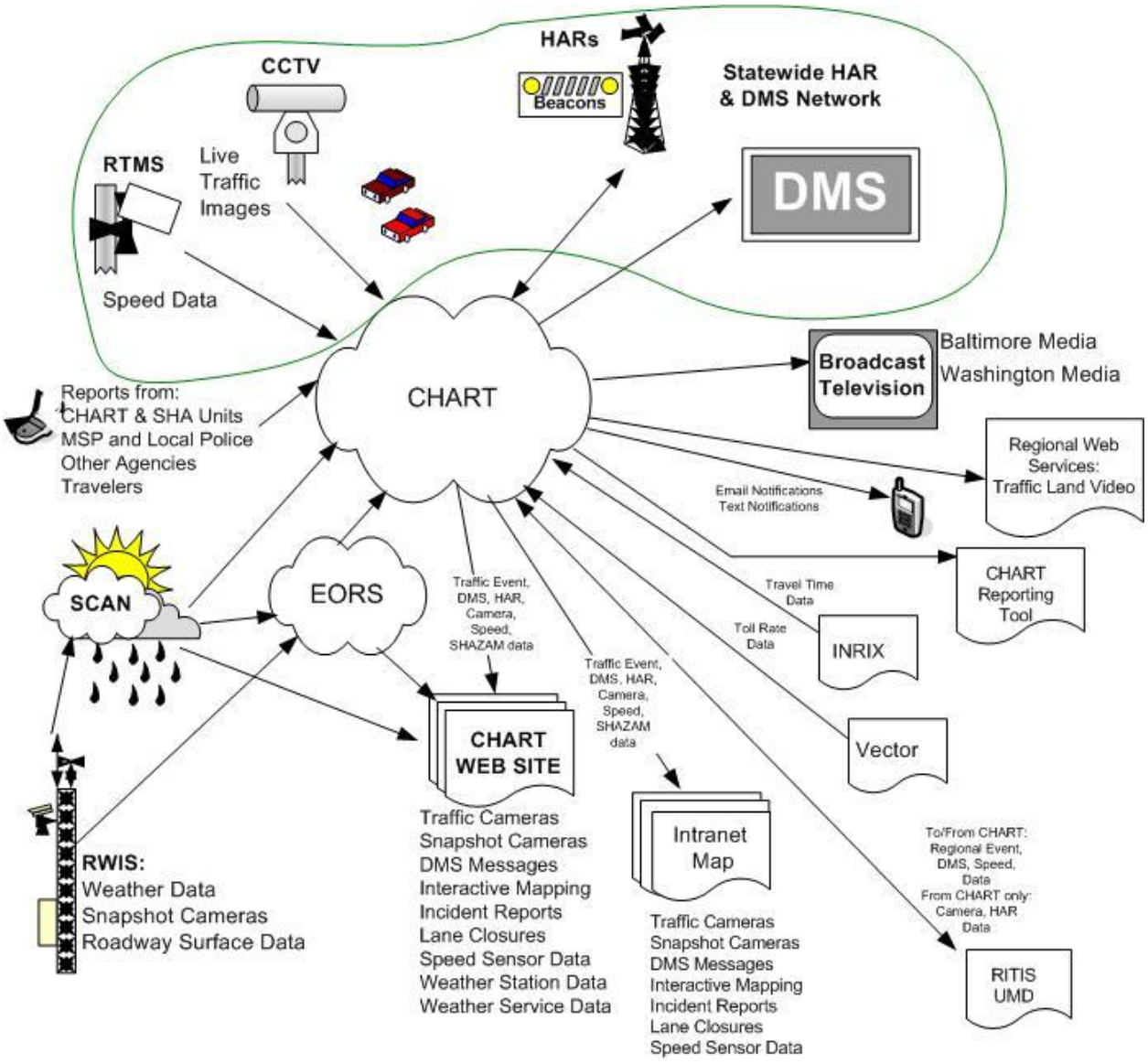


Figure 2-1 CHART and External Interfaces

The external interfaces modified for R8 are:

1. For R8, the interface between CHART and HARs (and SHAZAMs) shown at the top of the diagram is changed to include TCP/IP communications in addition to the existing Telephony communications. (Communication protocols are not shown on this diagram.)

Server and GUI deployment diagrams are shown in the next two figures. There are no changes to these diagrams for R8.

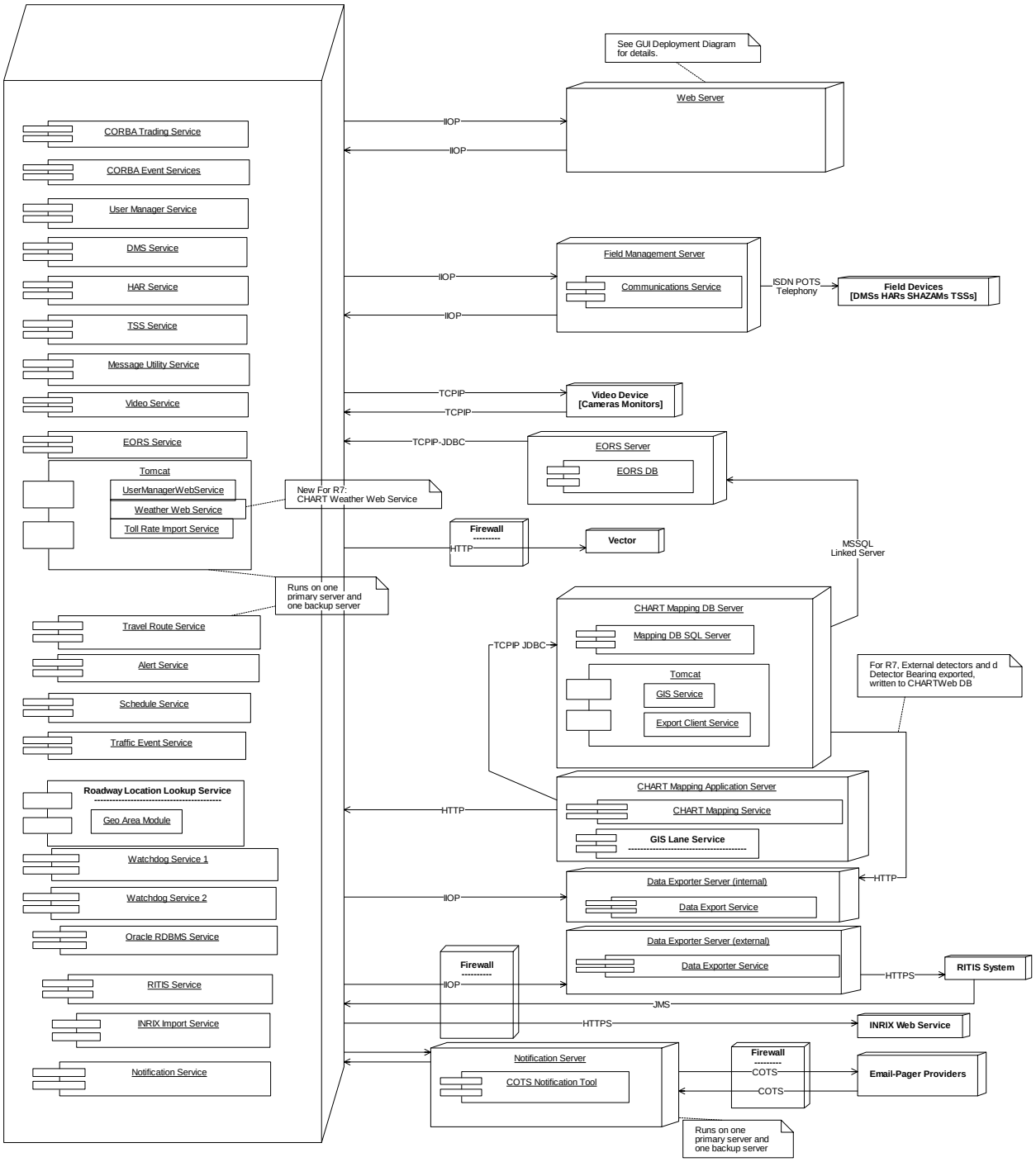


Figure 2-2 R8 Server Deployment

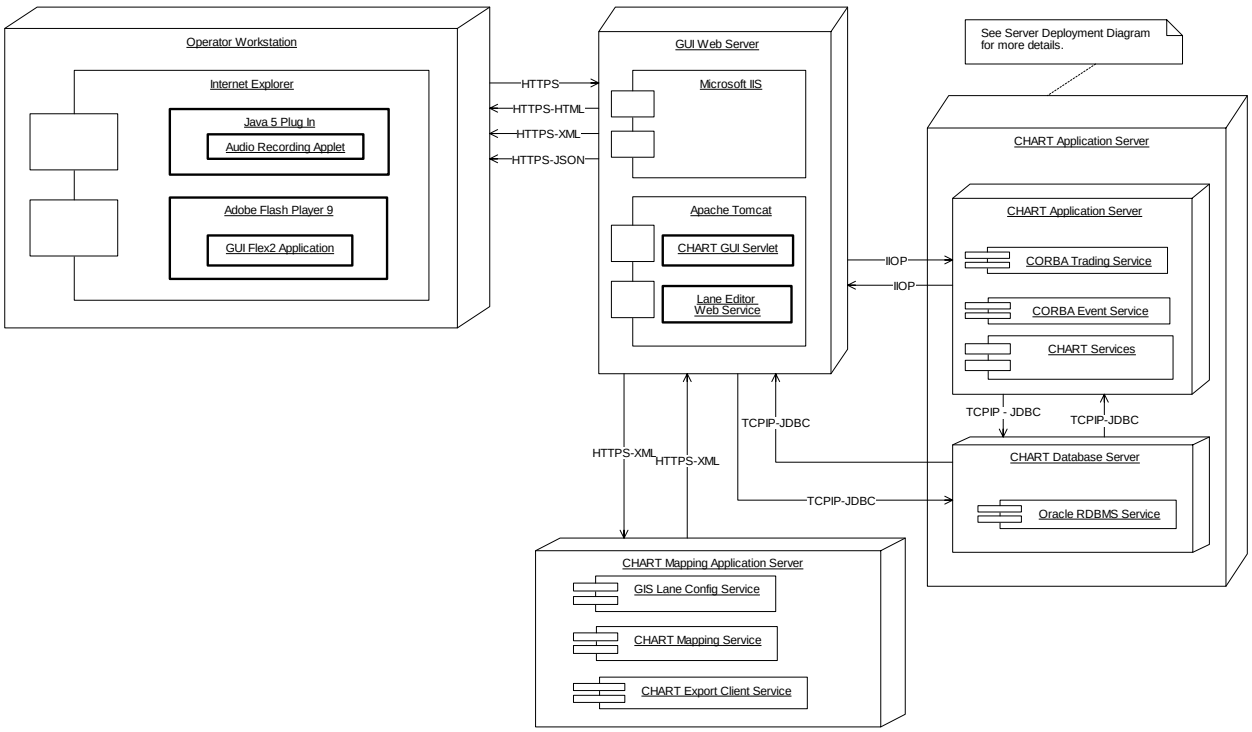


Figure 2-3 R8 GUI Deployment

2.2.2.1.2 Internal Interfaces

This section describes the internal interfaces being modified in Release 8 of the CHART system.

1. The existing GUI interface is changed for the IP HAR and IP SHAZAM features. The forms that are used to configure these devices and display details about these devices are changed. These changes are detailed in the Human Machine Interface section of this document. The CHART system IDL has been altered to allow the GUI to pass the new configuration information for the HARs and SHAZAMs to the HAR Service for persistence.

2.2.2.2 Mapping

There are no changes to the Mapping Application for this release.

2.3 Security

This section describes the security being added or modified in Release 8 of the CHART system. Unless otherwise noted, features being added for CHART Release 8 do not change security aspects of the CHART system.

2.4 Data

CHART Release 8 will be tested with the currently fielded Oracle database patches.

2.4.1 Data Storage

The CHART System stores most of its data in an Oracle database. Additionally the Integrated Map feature adds the ability to store location aliases to the spatial SQL Server database. Some data is stored in flat files on the CHART servers.

The Mapping Application stores and reads its data from a SQLServer datasbase.

This section describes all of these types of data.

2.4.1.1 Database

2.4.1.1.1 Database Architecture

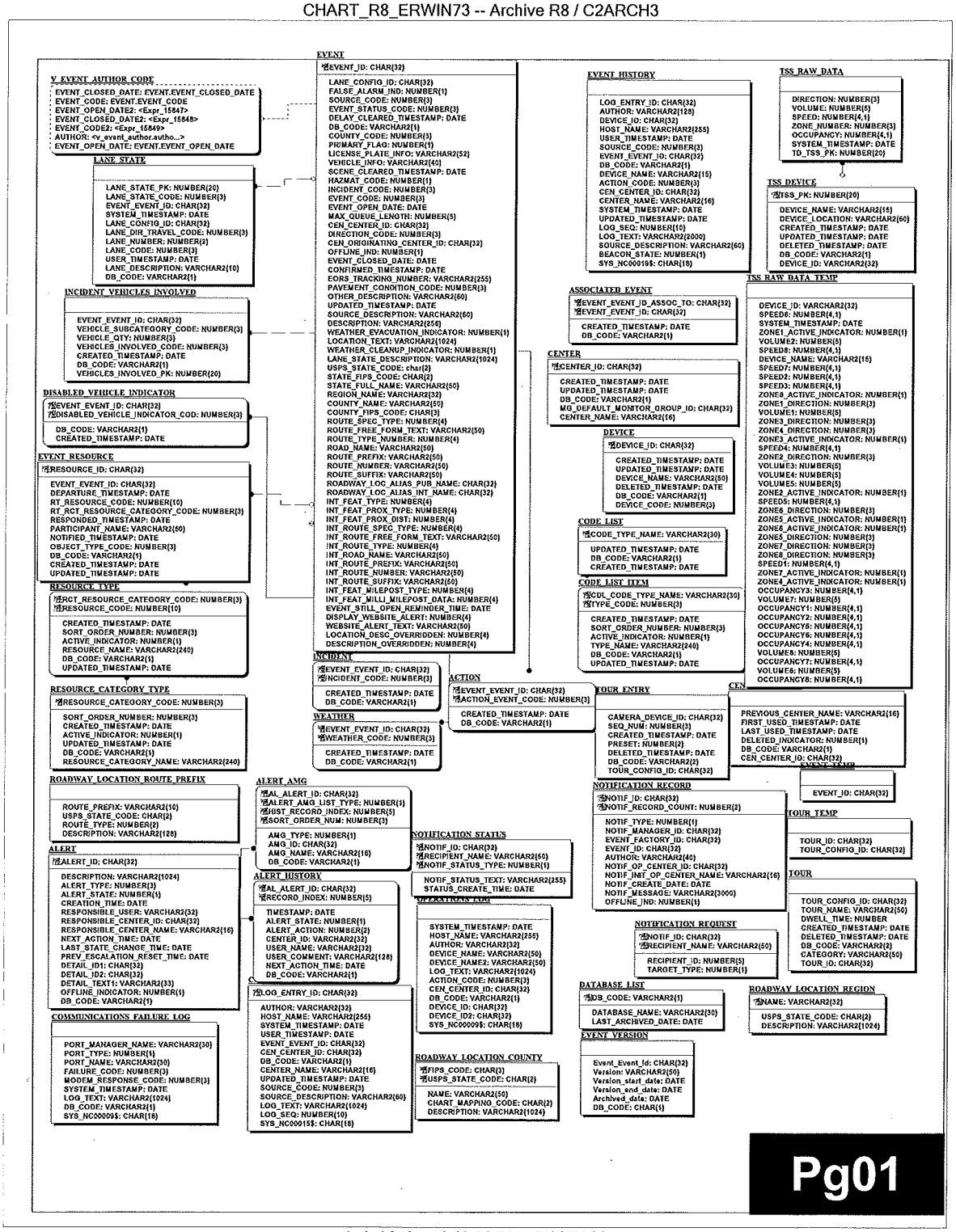
Except as noted CHART Release 8 features do not impact the overall architecture of the CHART database.

2.4.1.1.2 Logical Design

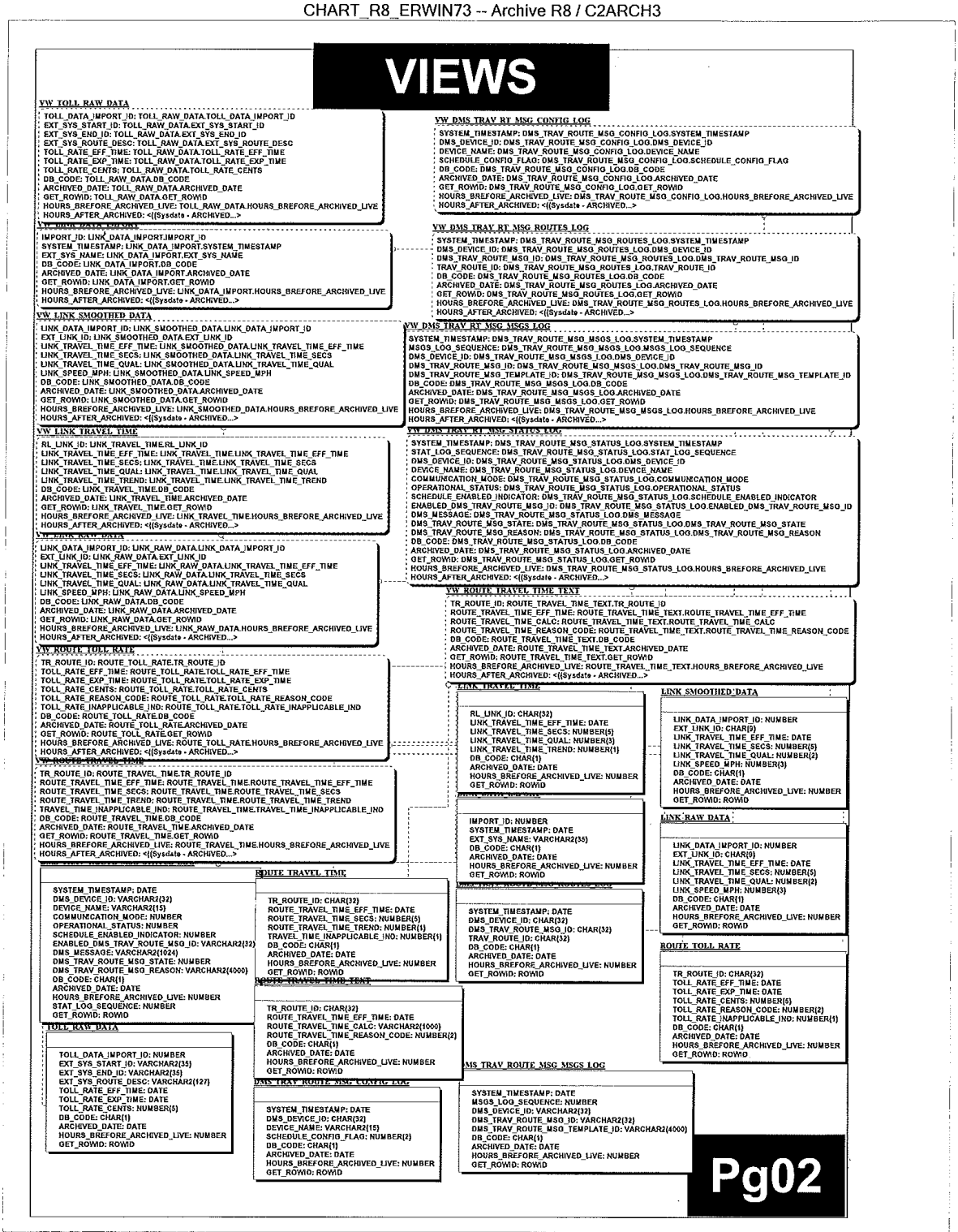
2.4.1.1.2.1 CHART Entity Relationship Diagram (ERD)

CHART Database entity relationship diagrams are shown below in the multiple pages of figures labeled collectively as Figure 2-5.

CHART_R8_ERWIN73 -- Archive R8 / C2ARCH3

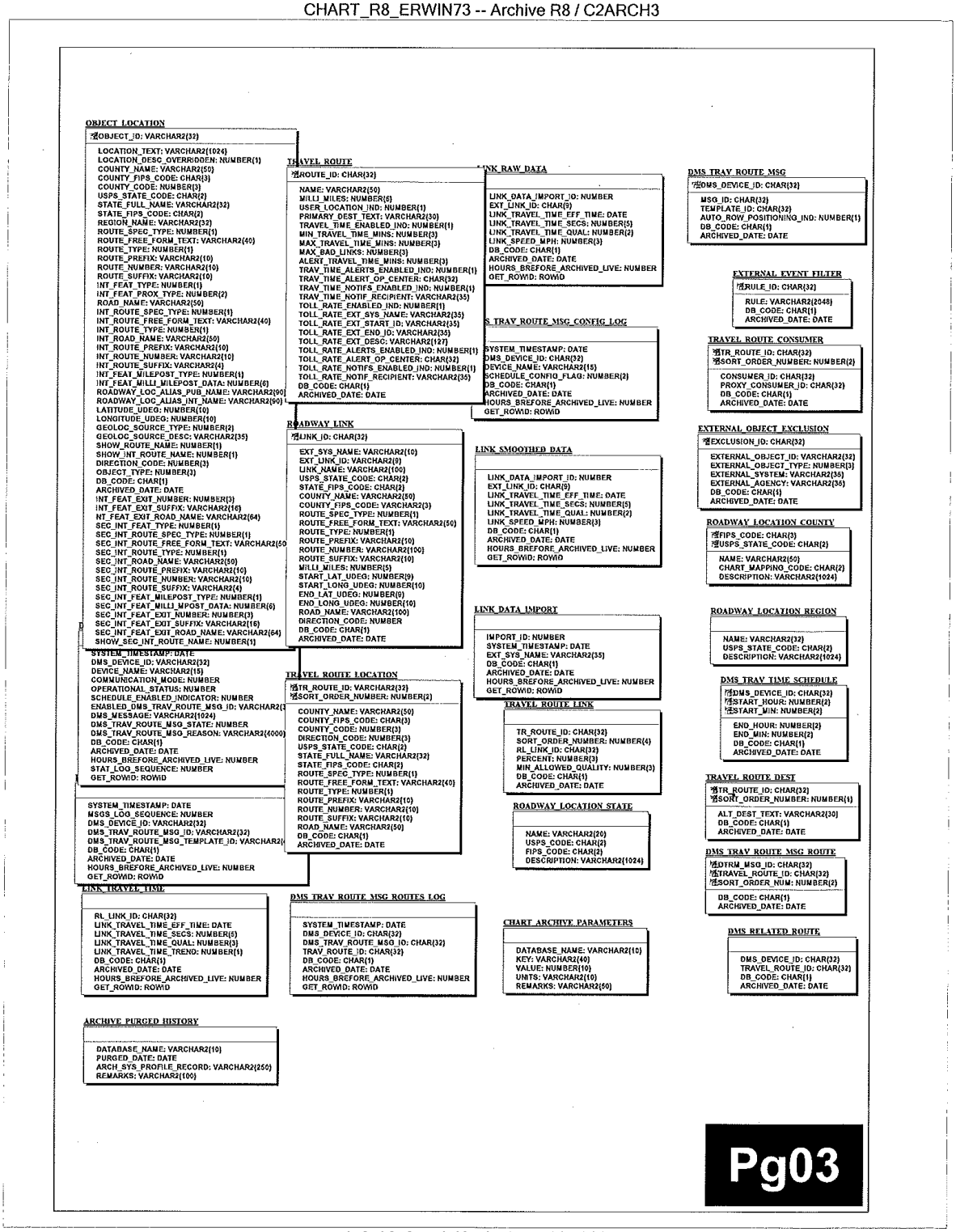


1, 1 / 2, 3 -- 4:43:59 PM , 5/11/2011



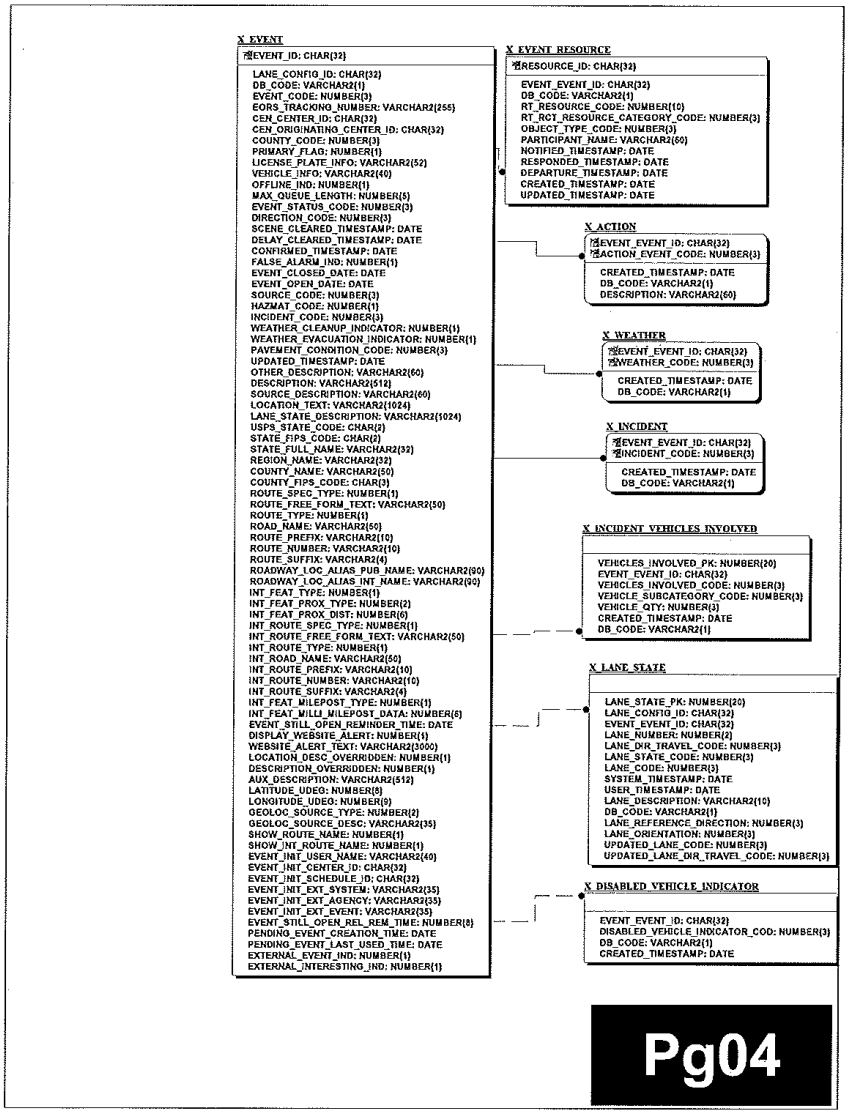
1, 2 / 2, 3 -- 4:43:59 PM , 5/11/2011

CHART_R8_ERWIN73 -- Archive R8 / C2ARCH3

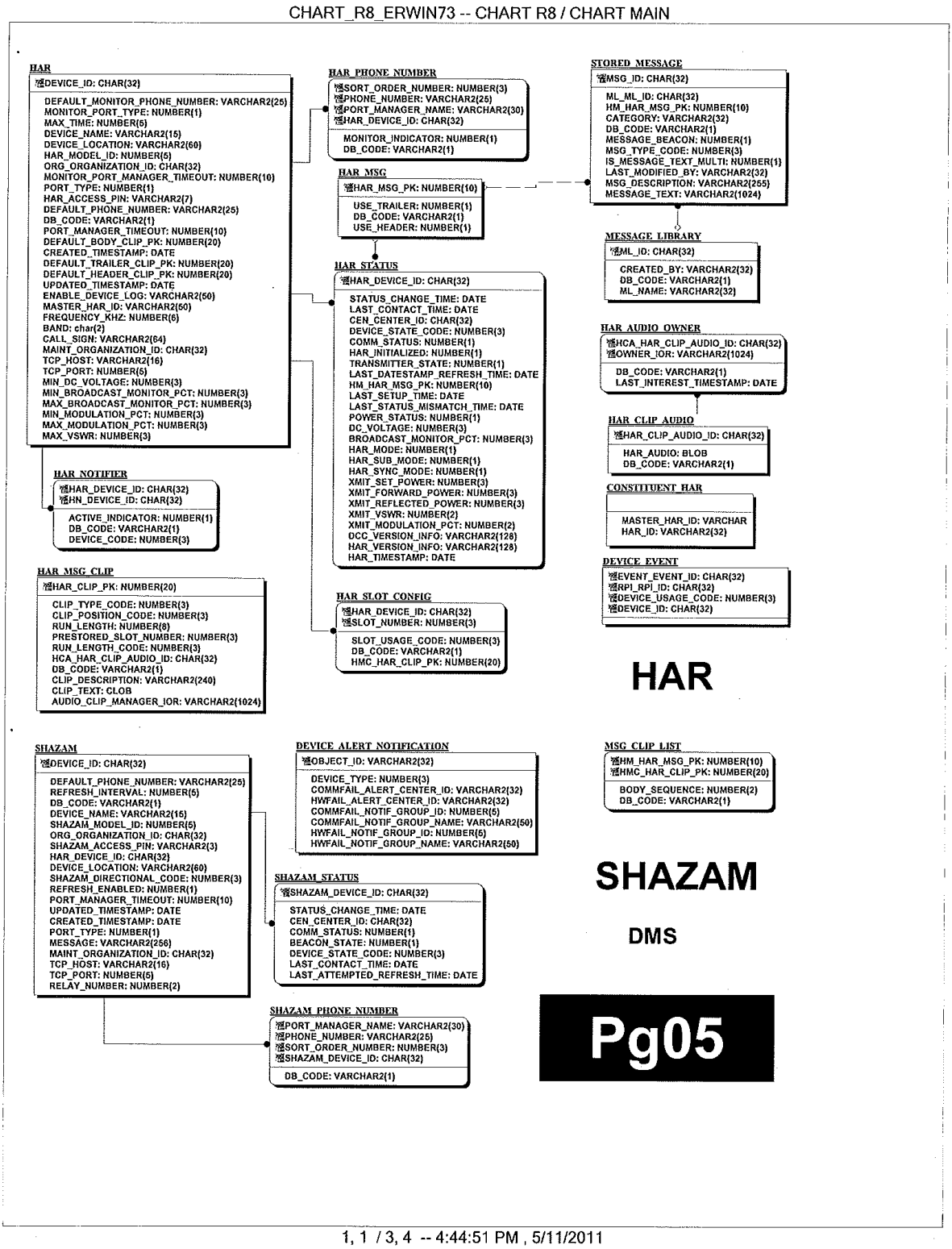


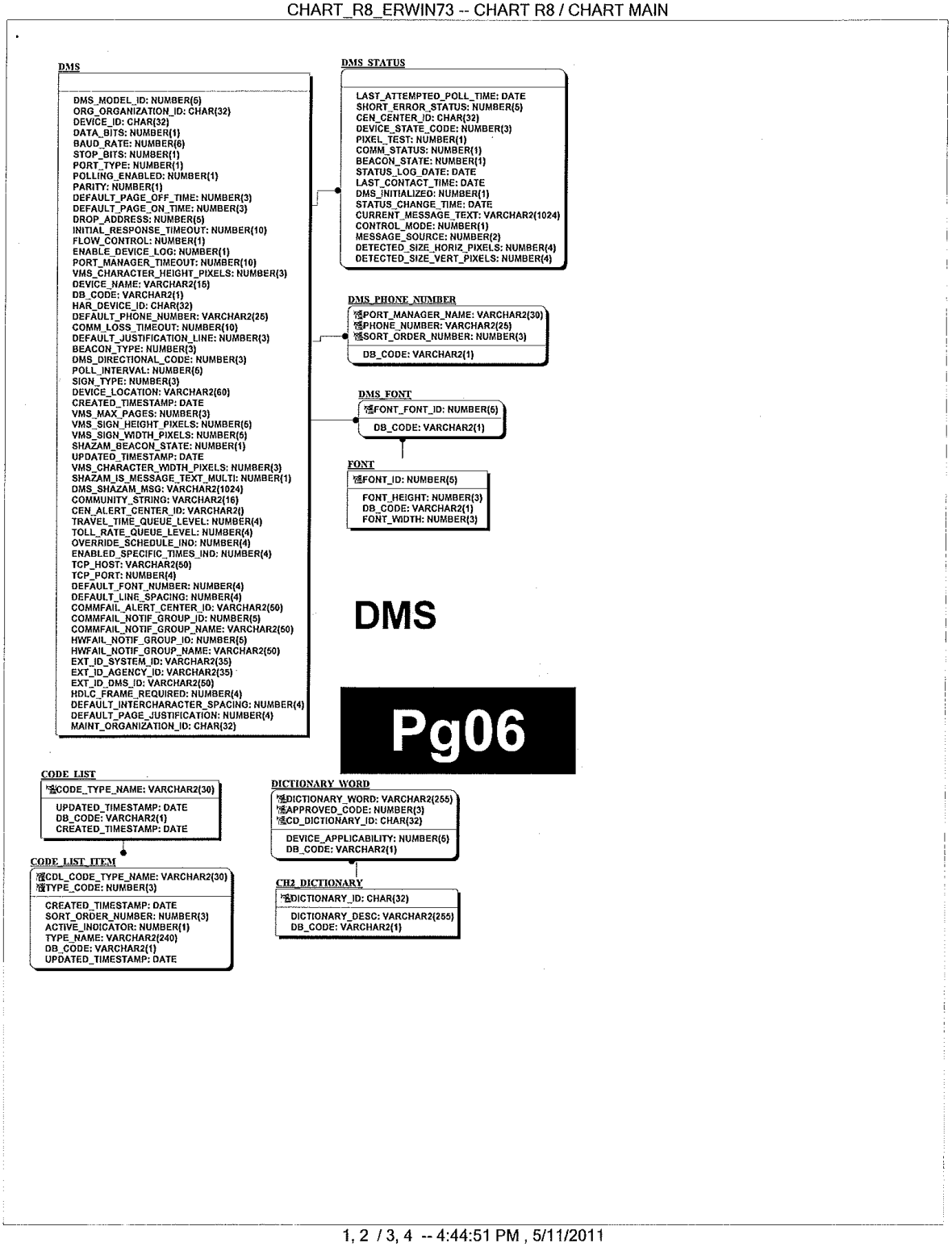
1, 3 / 2, 3 -- 4:43:59 PM , 5/11/2011

CHART_R8_ERWIN73 -- Archive R8 / C2ARCH3

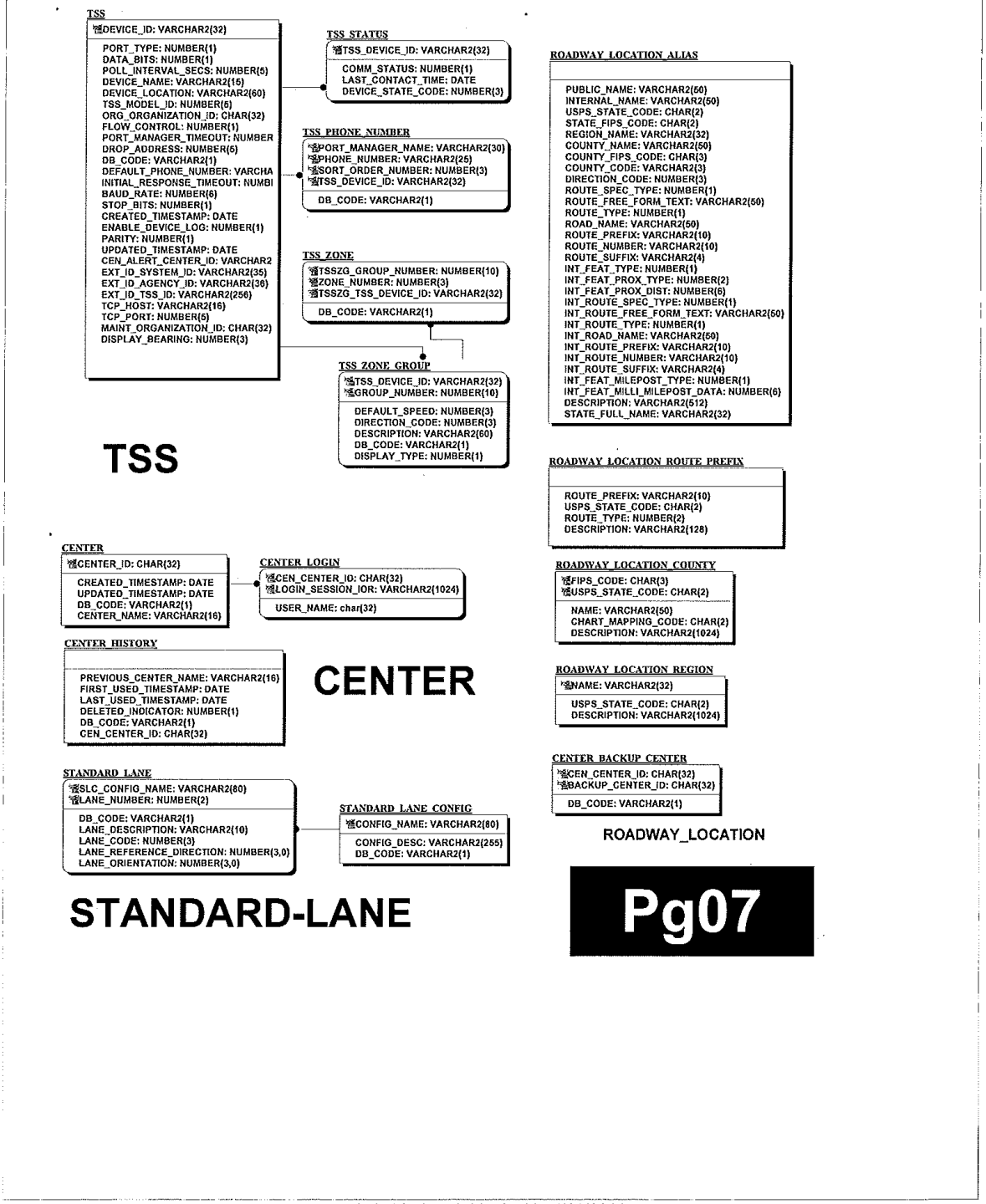


2, 1 / 2, 3 -- 4:43:59 PM , 5/11/2011


CHART_R8_ERWIN73 -- CHART R8 / CHART MAIN
HAR
DEVICE_ID: CHAR(32)
DEFAULT_MONITOR_PHONE_NUMBER: VARCHAR2(25)
MONITOR_PORT_TYPE: NUMBER(1)
MAX_TIME: NUMBER(5)
DEVICE_NAME: VARCHAR2(15)
DEVICE_LOCATION: VARCHAR2(60)
HAR_MODEL_ID: NUMBER(5)
ORG_ORGANIZATION_ID: CHAR(32)
MONITOR_PORT_MANAGER_TIMEOUT: NUMBER(10)
PORT_TYPE: NUMBER(1)
HAR_ACCESS_PIN: VARCHAR2(7)
DEFAULT_PHONE_NUMBER: VARCHAR2(25)
DB_CODE: VARCHAR2(1)
PORT_MANAGER_TIMEOUT: NUMBER(10)
DEFAULT_BODY_CLIP_PK: NUMBER(20)
CREATED_TIMESTAMP: DATE
DEFAULT_TRAILER_CLIP_PK: NUMBER(20)
DEFAULT_HEADER_CLIP_PK: NUMBER(20)
UPDATED_TIMESTAMP: DATE
ENABLE_DEVICE_LOG: VARCHAR2(50)
MASTER_HAR_ID: VARCHAR2(50)
FREQUENCY_KHZ: NUMBER(6)
BAND: char(2)
CALL_SIGN: VARCHAR2(64)
MAINT_ORGANIZATION_ID: CHAR(32)
TCP_HOST: VARCHAR2(16)
TCP_PORT: NUMBER(5)
MIN_DC_VOLTAGE: NUMBER(3)
MIN_BROADCAST_MONITOR_PCT: NUMBER(3)
MAX_BROADCAST_MONITOR_PCT: NUMBER(3)
MIN_MODULATION_PCT: NUMBER(3)
MAX_MODULATION_PCT: NUMBER(3)
MAX_VSWR: NUMBER(3)
HAR_PHONE_NUMBER
SORT_ORDER_NUMBER: NUMBER(3)
PHONE_NUMBER: VARCHAR2(25)
PORT_MANAGER_NAME: VARCHAR2(30)
HAR_DEVICE_ID: CHAR(32)
MONITOR_INDICATOR: NUMBER(1)
DB_CODE: VARCHAR2(1)
STORED_MESSAGE
MSG_ID: CHAR(32)
ML_ML_ID: CHAR(32)
HM_HAR_MSG_PK: NUMBER(10)
CATEGORY: VARCHAR2(32)
DB_CODE: VARCHAR2(1)
MESSAGE_BEACON: NUMBER(1)
MSG_TYPE_CODE: NUMBER(3)
IS_MESSAGE_TEXT_MULTI: NUMBER(1)
LAST_MODIFIED_BY: VARCHAR2(32)
MSG_DESCRIPTION: VARCHAR2(255)
MESSAGE_TEXT: VARCHAR2(1024)
HAR_MSG
HAR_MSG_PK: NUMBER(10)
USE_TRAILER: NUMBER(1)
DB_CODE: VARCHAR2(1)
USE_HEADER: NUMBER(1)
MESSAGE_LIBRARY
ML_ID: CHAR(32)
CREATED_BY: VARCHAR2(32)
DB_CODE: VARCHAR2(1)
ML_NAME: VARCHAR2(32)
HAR_STATUS
HAR_DEVICE_ID: CHAR(32)
STATUS_CHANGE_TIME: DATE
LAST_CONTACT_TIME: DATE
CEN_CENTER_ID: CHAR(32)
DEVICE_STATE_CODE: NUMBER(3)
COMM_STATUS: NUMBER(1)
HAR_INITIALIZED: NUMBER(1)
TRANSMITTER_STATE: NUMBER(1)
LAST_DATESTAMP_REFRESH_TIME: DATE
HM_HAR_MSG_PK: NUMBER(10)
LAST_SETUP_TIME: DATE
LAST_STATUS_MISMATCH_TIME: DATE
POWER_STATUS: NUMBER(1)
DC_VOLTAGE: NUMBER(3)
BROADCAST_MONITOR_PCT: NUMBER(3)
HAR_MODE: NUMBER(1)
HAR_SUB_MODE: NUMBER(1)
HAR_SYNC_MODE: NUMBER(1)
XMIT_SET_POWER: NUMBER(3)
XMIT_FORWARD_POWER: NUMBER(3)
XMIT_REFLECTED_POWER: NUMBER(3)
XMIT_VSWR: NUMBER(2)
XMIT_MODULATION_PCT: NUMBER(2)
OCC_VERSION_INFO: VARCHAR2(128)
HAR_VERSION_INFO: VARCHAR2(128)
HAR_TIMESTAMP: DATE
HAR_AUDIO_OWNER
HCA_HAR_CLIP_AUDIO_ID: CHAR(32)
OWNER_IOR: VARCHAR2(1024)
DB_CODE: VARCHAR2(1)
LAST_INTEREST_TIMESTAMP: DATE
HAR_CLIP_AUDIO
HAR_CLIP_AUDIO_ID: CHAR(32)
HAR_AUDIO: BLOB
DB_CODE: VARCHAR2(1)
CONSTITUENT_HAR
MASTER_HAR_ID: VARCHAR
HAR_ID: VARCHAR2(32)
DEVICE_EVENT
EVENT_EVENT_ID: CHAR(32)
RPI_RPI_ID: CHAR(32)
DEVICE_USAGE_CODE: NUMBER(3)
DEVICE_ID: CHAR(32)
HAR_NOTIFIER
HAR_DEVICE_ID: CHAR(32)
HN_DEVICE_ID: CHAR(32)
ACTIVE_INDICATOR: NUMBER(1)
DB_CODE: VARCHAR2(1)
DEVICE_CODE: NUMBER(3)
HAR_MSG_CLIP
HAR_CLIP_PK: NUMBER(20)
CLIP_TYPE_CODE: NUMBER(3)
CLIP_POSITION_CODE: NUMBER(3)
RUN_LENGTH: NUMBER(8)
PRESTORED_SLOT_NUMBER: NUMBER(3)
RUN_LENGTH_CODE: NUMBER(3)
HCA_HAR_CLIP_AUDIO_ID: CHAR(32)
DB_CODE: VARCHAR2(1)
CLIP_DESCRIPTION: VARCHAR2(240)
CLIP_TEXT: CLOB
AUDIO_CLIP_MANAGER_IOR: VARCHAR2(1024)
HAR_SLOT_CONFIG
HAR_DEVICE_ID: CHAR(32)
SLOT_NUMBER: NUMBER(3)
SLOT_USAGE_CODE: NUMBER(3)
DB_CODE: VARCHAR2(1)
HMC_HAR_CLIP_PK: NUMBER(20)
HAR
SHAZAM
DEVICE_ID: CHAR(32)
DEFAULT_PHONE_NUMBER: VARCHAR2(25)
REFRESH_INTERVAL: NUMBER(5)
DB_CODE: VARCHAR2(1)
DEVICE_NAME: VARCHAR2(15)
SHAZAM_MODEL_ID: NUMBER(5)
ORG_ORGANIZATION_ID: CHAR(32)
SHAZAM_ACCESS_PIN: VARCHAR2(3)
HAR_DEVICE_ID: CHAR(32)
DEVICE_LOCATION: VARCHAR2(60)
SHAZAM_DIRECTIONAL_CODE: NUMBER(3)
REFRESH_ENABLED: NUMBER(1)
PORT_MANAGER_TIMEOUT: NUMBER(10)
UPDATED_TIMESTAMP: DATE
CREATED_TIMESTAMP: DATE
PORT_TYPE: NUMBER(1)
MESSAGE: VARCHAR2(256)
MAINT_ORGANIZATION_ID: CHAR(32)
TCP_HOST: VARCHAR2(16)
TCP_PORT: NUMBER(5)
RELAY_NUMBER: NUMBER(2)
DEVICE_ALERT_NOTIFICATION
OBJECT_ID: VARCHAR2(32)
DEVICE_TYPE: NUMBER(3)
COMMFAIL_ALERT_CENTER_ID: VARCHAR2(32)
HWFAIL_ALERT_CENTER_ID: VARCHAR2(32)
COMMFAIL_NOTIF_GROUP_ID: NUMBER(5)
COMMFAIL_NOTIF_GROUP_NAME: VARCHAR2(50)
HWFAIL_NOTIF_GROUP_ID: NUMBER(5)
HWFAIL_NOTIF_GROUP_NAME: VARCHAR2(50)
SHAZAM_STATUS
SHAZAM_DEVICE_ID: CHAR(32)
STATUS_CHANGE_TIME: DATE
CEN_CENTER_ID: CHAR(32)
COMM_STATUS: NUMBER(1)
BEACON_STATE: NUMBER(1)
DEVICE_STATE_CODE: NUMBER(3)
LAST_CONTACT_TIME: DATE
LAST_ATTEMPTED_REFRESH_TIME: DATE
SHAZAM_PHONE_NUMBER
PORT_MANAGER_NAME: VARCHAR2(30)
PHONE_NUMBER: VARCHAR2(25)
SORT_ORDER_NUMBER: NUMBER(3)
SHAZAM_DEVICE_ID: CHAR(32)
DB_CODE: VARCHAR2(1)
MSG_CLIP_LIST
HM_HAR_MSG_PK: NUMBER(10)
HMC_HAR_CLIP_PK: NUMBER(20)
BODY_SEQUENCE: NUMBER(2)
DB_CODE: VARCHAR2(1)
SHAZAM
DMS
Pg05
1, 1 / 3, 4 -- 4:44:51 PM , 5/11/2011

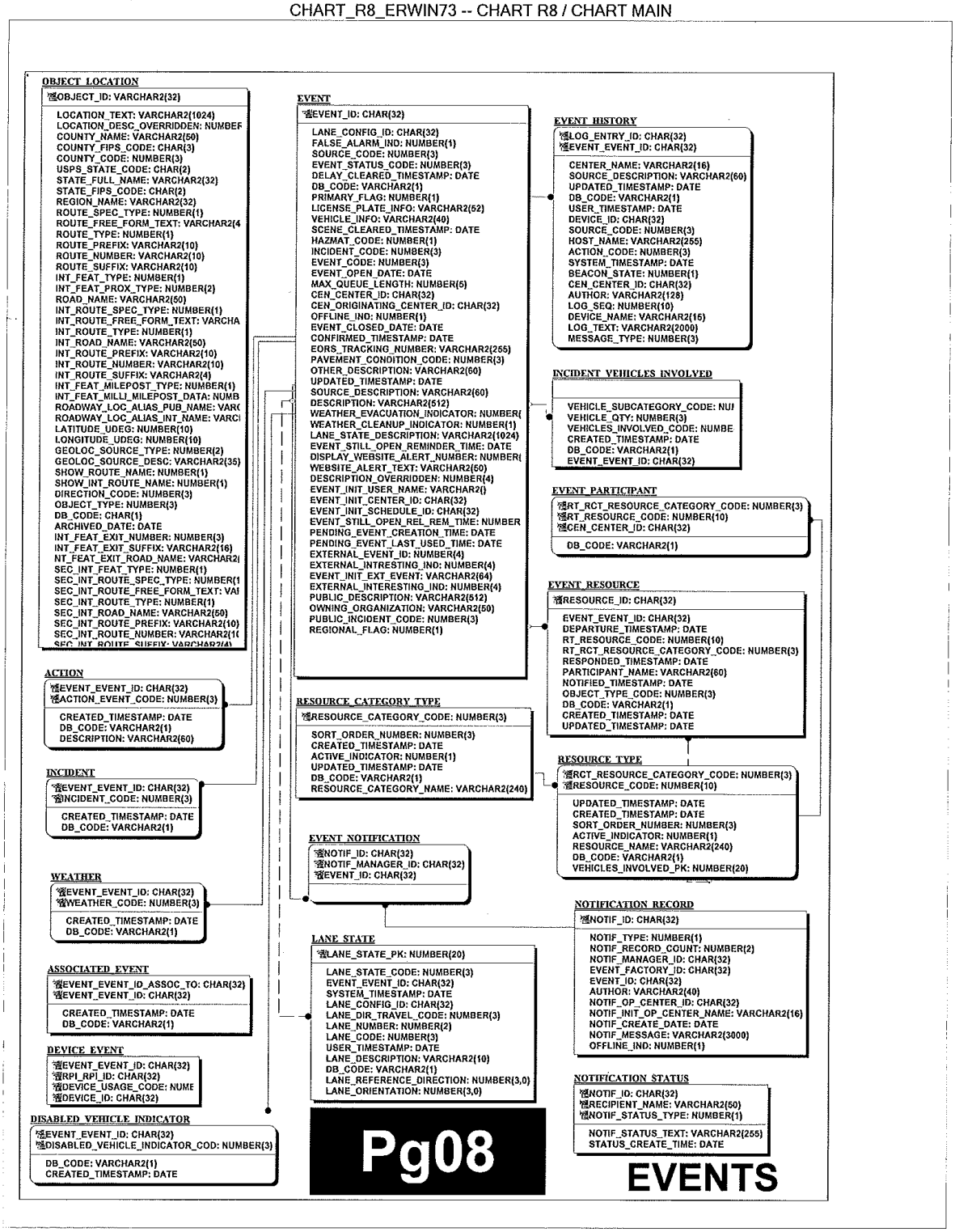

CHART_R8_ERWIN73 -- CHART R8 / CHART MAIN
DMS
DMS_MODEL_ID: NUMBER(5)
ORG_ORGANIZATION_ID: CHAR(32)
DEVICE_ID: CHAR(32)
DATA_BITS: NUMBER(1)
BAUD_RATE: NUMBER(6)
STOP_BITS: NUMBER(1)
PORT_TYPE: NUMBER(1)
POLLING_ENABLED: NUMBER(1)
PARITY: NUMBER(1)
DEFAULT_PAGE_OFF_TIME: NUMBER(3)
DEFAULT_PAGE_ON_TIME: NUMBER(3)
DROP_ADDRESS: NUMBER(5)
INITIAL_RESPONSE_TIMEOUT: NUMBER(10)
FLOW_CONTROL: NUMBER(1)
ENABLE_DEVICE_LOG: NUMBER(1)
PORT_MANAGER_TIMEOUT: NUMBER(10)
VMS_CHARACTER_HEIGHT_PIXELS: NUMBER(3)
DEVICE_NAME: VARCHAR2(15)
DB_CODE: VARCHAR2(1)
HAR_DEVICE_ID: CHAR(32)
DEFAULT_PHONE_NUMBER: VARCHAR2(25)
COMM_LOSS_TIMEOUT: NUMBER(10)
DEFAULT_JUSTIFICATION_LINE: NUMBER(3)
BEACON_TYPE: NUMBER(3)
DMS_DIRECTIONAL_CODE: NUMBER(3)
POLL_INTERVAL: NUMBER(6)
SIGN_TYPE: NUMBER(3)
DEVICE_LOCATION: VARCHAR2(60)
CREATED_TIMESTAMP: DATE
VMS_MAX_PAGES: NUMBER(3)
VMS_SIGN_HEIGHT_PIXELS: NUMBER(5)
VMS_SIGN_WIDTH_PIXELS: NUMBER(5)
SHAZAM_BEACON_STATE: NUMBER(1)
UPDATED_TIMESTAMP: DATE
VMS_CHARACTER_WDTH_PIXELS: NUMBER(3)
SHAZAM_IS_MESSAGE_TEXT_MULTI: NUMBER(1)
DMS_SHAZAM_MSG: VARCHAR2(1024)
COMMUNITY_STRING: VARCHAR2(16)
CEN_ALERT_CENTER_ID: VARCHAR2()
TRAVEL_TIME_QUEUE_LEVEL: NUMBER(4)
TOLL_RATE_QUEUE_LEVEL: NUMBER(4)
OVERRIDE_SCHEDULE_IND: NUMBER(4)
ENABLED_SPECIFIC_TIMES_IND: NUMBER(4)
TCP_HOST: VARCHAR2(50)
TCP_PORT: NUMBER(4)
DEFAULT_FONT_NUMBER: NUMBER(4)
DEFAULT_LINE_SPACING: NUMBER(4)
COMMFAIL_ALERT_CENTER_ID: VARCHAR2(50)
COMMFAIL_NOTIF_GROUP_ID: NUMBER(5)
COMMFAIL_NOTIF_GROUP_NAME: VARCHAR2(50)
HWFAIL_NOTIF_GROUP_ID: NUMBER(5)
HWFAIL_NOTIF_GROUP_NAME: VARCHAR2(50)
EXT_ID_SYSTEM_ID: VARCHAR2(35)
EXT_ID_AGENCY_ID: VARCHAR2(35)
EXT_ID_DMS_ID: VARCHAR2(50)
HDLC_FRAME_REQUIRED: NUMBER(4)
DEFAULT_INTERCHARACTER_SPACING: NUMBER(4)
DEFAULT_PAGE_JUSTIFICATION: NUMBER(4)
MAINT_ORGANIZATION_ID: CHAR(32)
DMS_STATUS
LAST_ATTEMPTED_POLL_TIME: DATE
SHORT_ERROR_STATUS: NUMBER(5)
CEN_CENTER_ID: CHAR(32)
DEVICE_STATE_CODE: NUMBER(3)
PIXEL_TEST: NUMBER(1)
COMM_STATUS: NUMBER(1)
BEACON_STATE: NUMBER(1)
STATUS_LOG_DATE: DATE
LAST_CONTACT_TIME: DATE
DMS_INITIALIZED: NUMBER(1)
STATUS_CHANGE_TIME: DATE
CURRENT_MESSAGE_TEXT: VARCHAR2(1024)
CONTROL_MODE: NUMBER(1)
MESSAGE_SOURCE: NUMBER(2)
DETECTED_SIZE_HORIZ_PIXELS: NUMBER(4)
DETECTED_SIZE_VERT_PIXELS: NUMBER(4)
DMS_PHONE_NUMBER
PORT_MANAGER_NAME: VARCHAR2(30)
PHONE_NUMBER: VARCHAR2(25)
SORT_ORDER_NUMBER: NUMBER(3)
DB_CODE: VARCHAR2(1)
DMS_FONT
FONT_FONT_ID: NUMBER(5)
DB_CODE: VARCHAR2(1)
FONT
FONT_ID: NUMBER(5)
FONT_HEIGHT: NUMBER(3)
DB_CODE: VARCHAR2(1)
FONT_WIDTH: NUMBER(3)
DMS
Pg06
CODE_LIST
CODE_TYPE_NAME: VARCHAR2(30)
UPDATED_TIMESTAMP: DATE
DB_CODE: VARCHAR2(1)
CREATED_TIMESTAMP: DATE
CODE_LIST_ITEM
CDL_CODE_TYPE_NAME: VARCHAR2(30)
TYPE_CODE: NUMBER(3)
CREATED_TIMESTAMP: DATE
SORT_ORDER_NUMBER: NUMBER(3)
ACTIVE_INDICATOR: NUMBER(1)
TYPE_NAME: VARCHAR2(240)
DB_CODE: VARCHAR2(1)
UPDATED_TIMESTAMP: DATE
DICTIONARY_WORD
DICTIONARY_WORD: VARCHAR2(255)
APPROVED_CODE: NUMBER(3)
CD_DICTIONARY_ID: CHAR(32)
DEVICE_APPLICABILITY: NUMBER(5)
DB_CODE: VARCHAR2(1)
CH2_DICTIONARY
DICTIONARY_ID: CHAR(32)
DICTIONARY_DESC: VARCHAR2(255)
DB_CODE: VARCHAR2(1)
1, 2 / 3, 4 -- 4:44:51 PM , 5/11/2011

CHART_R8_ERWIN73 -- CHART R8 / CHART MAIN



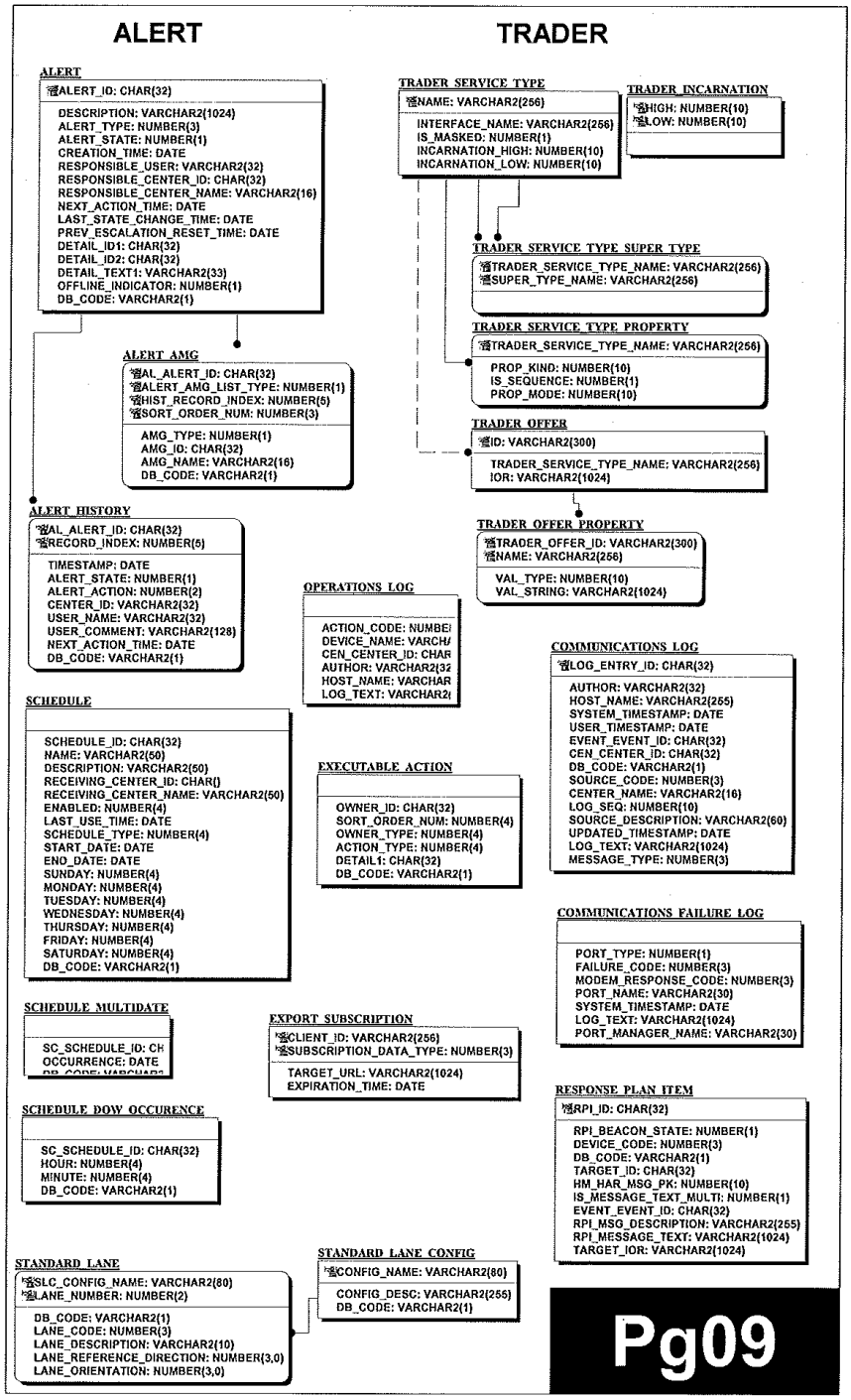
1, 3 / 3, 4 -- 4:44:51 PM , 5/11/2011

CHART_R8_ERWIN73 -- CHART R8 / CHART MAIN

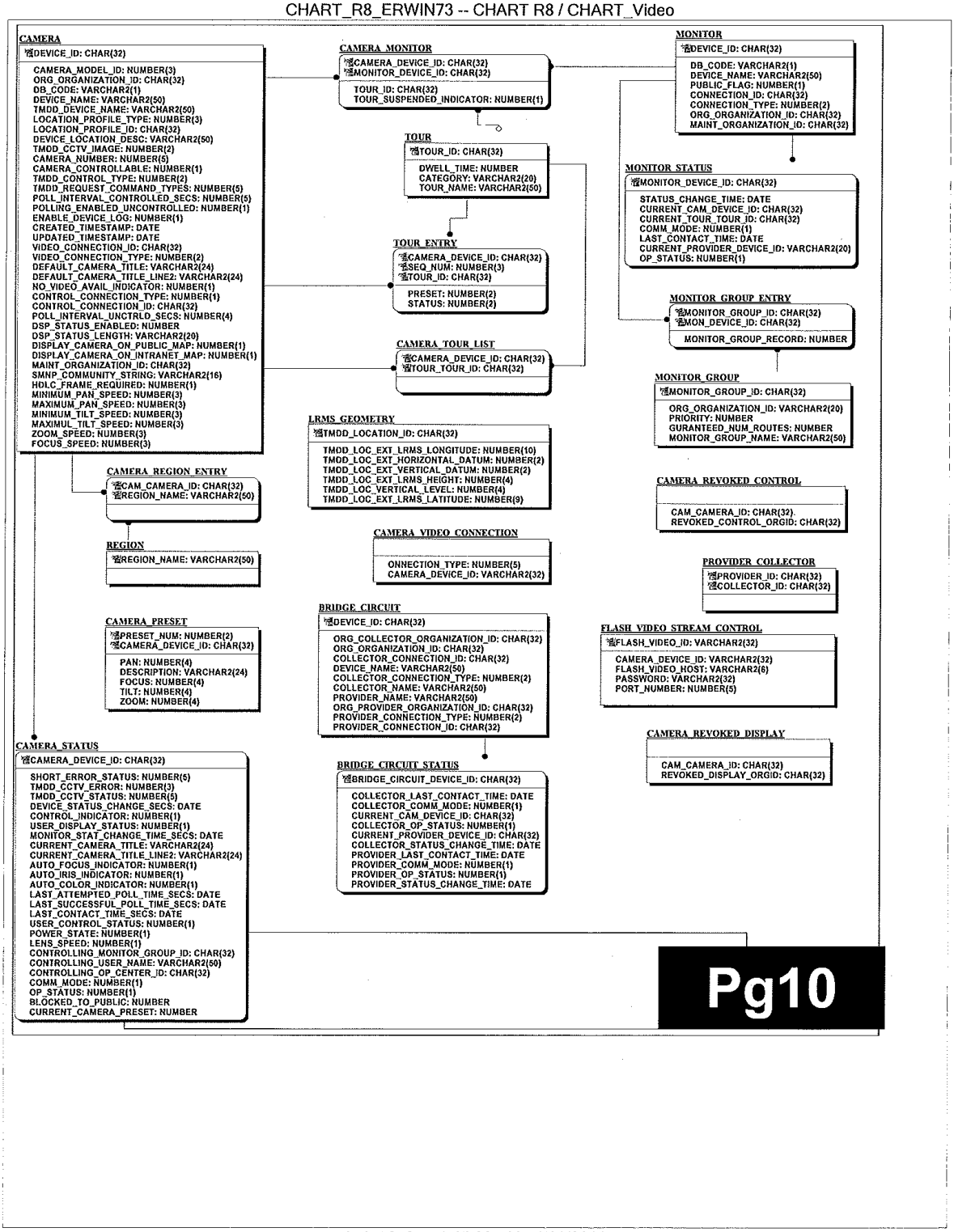


2, 1 / 3, 4 -- 4:44:51 PM , 5/11/2011

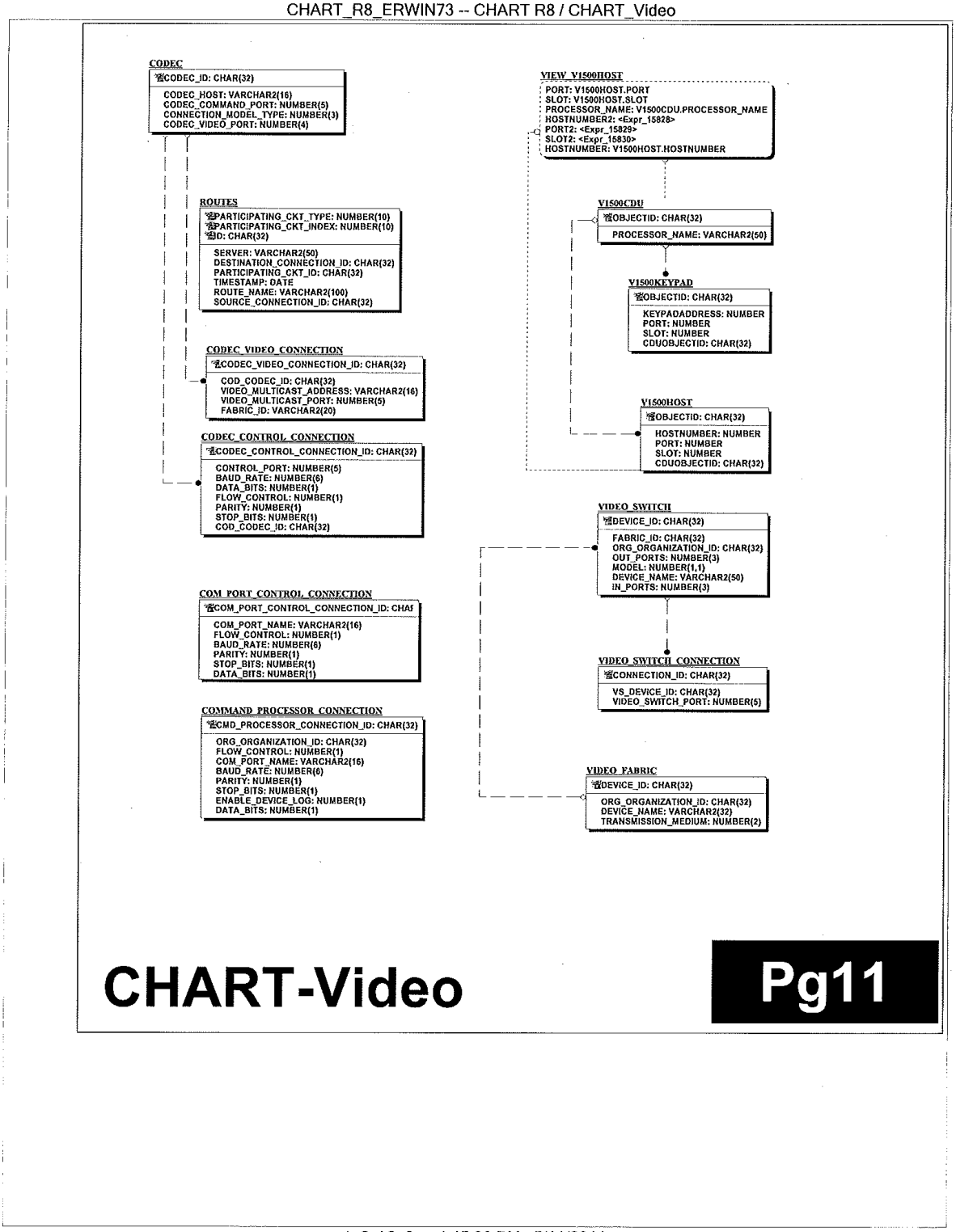
CHART_R8_ERWIN73 -- CHART R8 / CHART MAIN



2, 2 / 3, 4 -- 4:44:51 PM , 5/11/2011

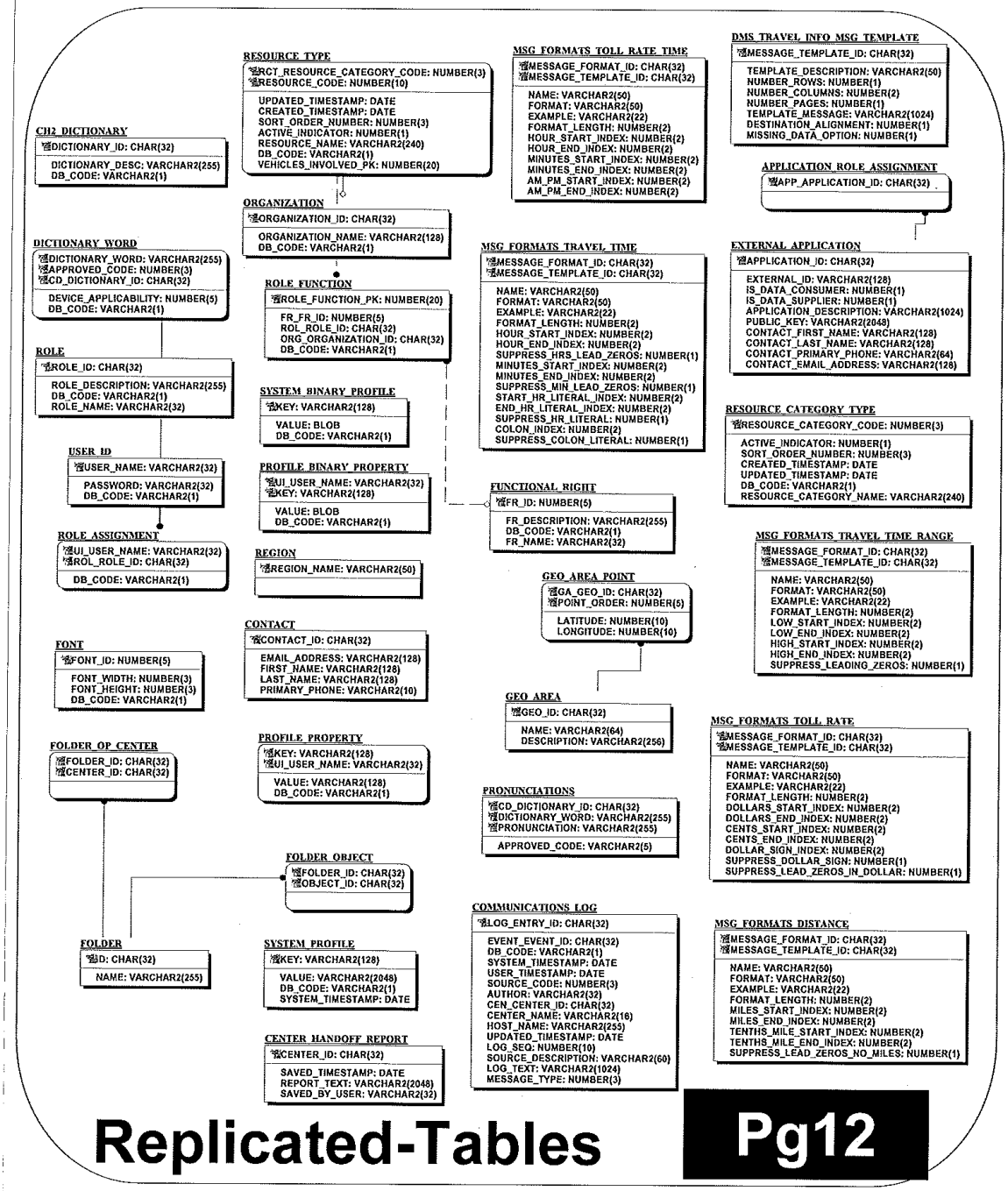

CHART_R8_ERWIN73 -- CHART R8 / CHART_Video
CAMERA
DEVICE_ID: CHAR(32)
CAMERA_MODEL_ID: NUMBER(3)
ORG_ORGANIZATION_ID: CHAR(32)
DB_CODE: VARCHAR2(1)
DEVICE_NAME: VARCHAR2(50)
TMDD_DEVICE_NAME: VARCHAR2(50)
LOCATION_PROFILE_TYPE: NUMBER(3)
LOCATION_PROFILE_ID: CHAR(32)
DEVICE_LOCATION_DESC: VARCHAR2(50)
TMDD_CCTV_IMAGE: NUMBER(2)
CAMERA_NUMBER: NUMBER(5)
CAMERA_CONTROLLABLE: NUMBER(1)
TMDD_CONTROL_TYPE: NUMBER(2)
TMDD_REQUEST_COMMAND_TYPES: NUMBER(5)
POLL_INTERVAL_CONTROLLED_SECS: NUMBER(5)
POLLING_ENABLED_UNCONTROLLED: NUMBER(1)
ENABLE_DEVICE_LOG: NUMBER(1)
CREATED_TIMESTAMP: DATE
UPDATED_TIMESTAMP: DATE
VIDEO_CONNECTION_ID: CHAR(32)
VIDEO_CONNECTION_TYPE: NUMBER(2)
DEFAULT_CAMERA_TITLE: VARCHAR2(24)
DEFAULT_CAMERA_TITLE_LINE2: VARCHAR2(24)
NO_VIDEO_AVAIL_INDICATOR: NUMBER(1)
CONTROL_CONNECTION_TYPE: NUMBER(1)
CONTROL_CONNECTION_ID: CHAR(32)
POLL_INTERVAL_UNCTRLD_SECS: NUMBER(4)
DSP_STATUS_ENABLED: NUMBER
DSP_STATUS_LENGTH: VARCHAR2(20)
DISPLAY_CAMERA_ON_PUBLIC_MAP: NUMBER(1)
DISPLAY_CAMERA_ON_INTRANET_MAP: NUMBER(1)
MAINT_ORGANIZATION_ID: CHAR(32)
SMNP_COMMUNITY_STRING: VARCHAR2(16)
HDLC_FRAME_REQUIRED: NUMBER(1)
MINIMUM_PAN_SPEED: NUMBER(3)
MAXIMUM_PAN_SPEED: NUMBER(3)
MINIMUM_TILT_SPEED: NUMBER(3)
MAXIMUL_TILT_SPEED: NUMBER(3)
ZOOM_SPEED: NUMBER(3)
FOCUS_SPEED: NUMBER(3)
CAMERA_MONITOR
CAMERA_DEVICE_ID: CHAR(32)
MONITOR_DEVICE_ID: CHAR(32)
TOUR_ID: CHAR(32)
TOUR_SUSPENDED_INDICATOR: NUMBER(1)
MONITOR
DEVICE_ID: CHAR(32)
DB_CODE: VARCHAR2(1)
DEVICE_NAME: VARCHAR2(50)
PUBLIC_FLAG: NUMBER(1)
CONNECTION_ID: CHAR(32)
CONNECTION_TYPE: NUMBER(2)
ORG_ORGANIZATION_ID: CHAR(32)
MAINT_ORGANIZATION_ID: CHAR(32)
TOUR
TOUR_ID: CHAR(32)
DWELL_TIME: NUMBER
CATEGORY: VARCHAR2(20)
TOUR_NAME: VARCHAR2(50)
MONITOR_STATUS
MONITOR_DEVICE_ID: CHAR(32)
STATUS_CHANGE_TIME: DATE
CURRENT_CAM_DEVICE_ID: CHAR(32)
CURRENT_TOUR_TOUR_ID: CHAR(32)
COMM_MODE: NUMBER(1)
LAST_CONTACT_TIME: DATE
CURRENT_PROVIDER_DEVICE_ID: VARCHAR2(20)
OP_STATUS: NUMBER(1)
TOUR_ENTRY
CAMERA_DEVICE_ID: CHAR(32)
SEQ_NUM: NUMBER(3)
TOUR_ID: CHAR(32)
PRESET: NUMBER(2)
STATUS: NUMBER(2)
MONITOR_GROUP_ENTRY
MONITOR_GROUP_ID: CHAR(32)
MON_DEVICE_ID: CHAR(32)
MONITOR_GROUP_RECORD: NUMBER
CAMERA_TOUR_LIST
CAMERA_DEVICE_ID: CHAR(32)
TOUR_TOUR_ID: CHAR(32)
MONITOR_GROUP
MONITOR_GROUP_ID: CHAR(32)
ORG_ORGANIZATION_ID: VARCHAR2(20)
PRIORITY: NUMBER
GURANTEED_NUM_ROUTES: NUMBER
MONITOR_GROUP_NAME: VARCHAR2(50)
LRMS_GEOMETRY
TMDD_LOCATION_ID: CHAR(32)
TMDD_LOC_EXT_LRMS_LONGITUDE: NUMBER(10)
TMDD_LOC_EXT_HORIZONTAL_DATUM: NUMBER(2)
TMDD_LOC_EXT_VERTICAL_DATUM: NUMBER(2)
TMDD_LOC_EXT_LRMS_HEIGHT: NUMBER(4)
TMDD_LOC_VERTICAL_LEVEL: NUMBER(4)
TMDD_LOC_EXT_LRMS_LATITUDE: NUMBER(9)
CAMERA_REGION_ENTRY
CAM_CAMERA_ID: CHAR(32)
REGION_NAME: VARCHAR2(50)
CAMERA_REVOKED_CONTROL
CAM_CAMERA_ID: CHAR(32)
REVOKED_CONTROL_ORGID: CHAR(32)
REGION
REGION_NAME: VARCHAR2(50)
CAMERA_VIDEO_CONNECTION
ONNECTION_TYPE: NUMBER(5)
CAMERA_DEVICE_ID: VARCHAR2(32)
PROVIDER_COLLECTOR
PROVIDER_ID: CHAR(32)
COLLECTOR_ID: CHAR(32)
CAMERA_PRESET
PRESET_NUM: NUMBER(2)
CAMERA_DEVICE_ID: CHAR(32)
PAN: NUMBER(4)
DESCRIPTION: VARCHAR2(24)
FOCUS: NUMBER(4)
TILT: NUMBER(4)
ZOOM: NUMBER(4)
BRIDGE_CIRCUIT
DEVICE_ID: CHAR(32)
ORG_COLLECTOR_ORGANIZATION_ID: CHAR(32)
ORG_ORGANIZATION_ID: CHAR(32)
COLLECTOR_CONNECTION_ID: CHAR(32)
DEVICE_NAME: VARCHAR2(50)
COLLECTOR_CONNECTION_TYPE: NUMBER(2)
COLLECTOR_NAME: VARCHAR2(50)
PROVIDER_NAME: VARCHAR2(50)
ORG_PROVIDER_ORGANIZATION_ID: CHAR(32)
PROVIDER_CONNECTION_TYPE: NUMBER(2)
PROVIDER_CONNECTION_ID: CHAR(32)
FLASH_VIDEO_STREAM_CONTROL
FLASH_VIDEO_ID: VARCHAR2(32)
CAMERA_DEVICE_ID: VARCHAR2(32)
FLASH_VIDEO_HOST: VARCHAR2(6)
PASSWORD: VARCHAR2(32)
PORT_NUMBER: NUMBER(5)
CAMERA_STATUS
CAMERA_DEVICE_ID: CHAR(32)
SHORT_ERROR_STATUS: NUMBER(5)
TMDD_CCTV_ERROR: NUMBER(3)
TMDD_CCTV_STATUS: NUMBER(5)
DEVICE_STATUS_CHANGE_SECS: DATE
CONTROL_INDICATOR: NUMBER(1)
USER_DISPLAY_STATUS: NUMBER(1)
MONITOR_STAT_CHANGE_TIME_SECS: DATE
CURRENT_CAMERA_TITLE: VARCHAR2(24)
CURRENT_CAMERA_TITLE_LINE2: VARCHAR2(24)
AUTO_FOCUS_INDICATOR: NUMBER(1)
AUTO_IRIS_INDICATOR: NUMBER(1)
AUTO_COLOR_INDICATOR: NUMBER(1)
LAST_ATTEMPTED_POLL_TIME_SECS: DATE
LAST_SUCCESSFUL_POLL_TIME_SECS: DATE
LAST_CONTACT_TIME_SECS: DATE
USER_CONTROL_STATUS: NUMBER(1)
POWER_STATE: NUMBER(1)
LENS_SPEED: NUMBER(1)
CONTROLLING_MONITOR_GROUP_ID: CHAR(32)
CONTROLLING_USER_NAME: VARCHAR2(50)
CONTROLLING_OP_CENTER_ID: CHAR(32)
COMM_MODE: NUMBER(1)
OP_STATUS: NUMBER(1)
BLOCKED_TO_PUBLIC: NUMBER
CURRENT_CAMERA_PRESET: NUMBER
BRIDGE_CIRCUIT_STATUS
BRIDGE_CIRCUIT_DEVICE_ID: CHAR(32)
COLLECTOR_LAST_CONTACT_TIME: DATE
COLLECTOR_COMM_MODE: NUMBER(1)
CURRENT_CAM_DEVICE_ID: CHAR(32)
COLLECTOR_OP_STATUS: NUMBER(1)
CURRENT_PROVIDER_DEVICE_ID: CHAR(32)
COLLECTOR_STATUS_CHANGE_TIME: DATE
PROVIDER_LAST_CONTACT_TIME: DATE
PROVIDER_COMM_MODE: NUMBER(1)
PROVIDER_OP_STATUS: NUMBER(1)
PROVIDER_STATUS_CHANGE_TIME: DATE
CAMERA_REVOKED_DISPLAY
CAM_CAMERA_ID: CHAR(32)
REVOKED_DISPLAY_ORGID: CHAR(32)
Pg10
1, 1 / 2, 3 -- 4:45:00 PM , 5/11/2011

CHART_R8_ERWIN73 -- CHART R8 / CHART_Video

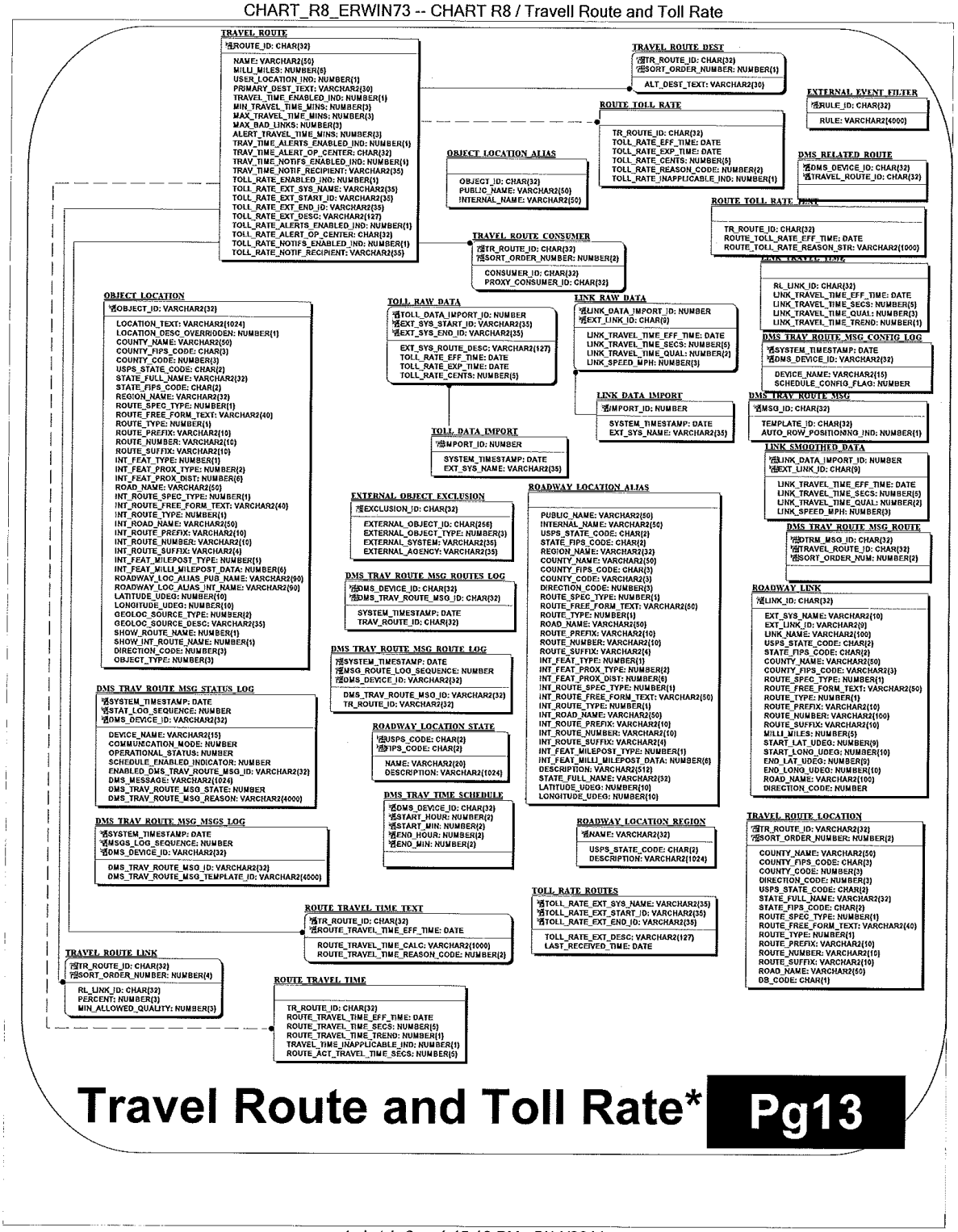


1, 2 / 2, 3 -- 4:45:00 PM , 5/11/2011

CHART_R8_ERWIN73 -- CHART R8 / Replicated Tables



1, 1 / 1, 1 -- 4:45:09 PM , 5/11/2011


CHART_R8_ERWIN73 -- CHART R8 / Travell Route and Toll Rate
TRAVEL ROUTE
*ROUTE_ID: CHAR(32)
NAME: VARCHAR2(50)
MILLI_MILES: NUMBER(5)
USER_LOCATION_IND: NUMBER(1)
PRIMARY_DEST_TEXT: VARCHAR2(30)
TRAVEL_TIME_ENABLED_IND: NUMBER(1)
MIN_TRAVEL_TIME_MINS: NUMBER(3)
MAX_TRAVEL_TIME_MINS: NUMBER(3)
MAX_BAD_LINKS: NUMBER(3)
ALERT_TRAVEL_TIME_MINS: NUMBER(3)
TRAV_TIME_ALERTS_ENABLED_IND: NUMBER(1)
TRAV_TIME_ALERT_OP_CENTER: CHAR(32)
TRAV_TIME_NOTIFS_ENABLED_IND: NUMBER(1)
TRAV_TIME_NOTIF_RECIPIENT: VARCHAR2(35)
TOLL_RATE_ENABLED_IND: NUMBER(1)
TOLL_RATE_EXT_SYS_NAME: VARCHAR2(35)
TOLL_RATE_EXT_START_ID: VARCHAR2(35)
TOLL_RATE_EXT_END_ID: VARCHAR2(35)
TOLL_RATE_EXT_DESC: VARCHAR2(127)
TOLL_RATE_ALERTS_ENABLED_IND: NUMBER(1)
TOLL_RATE_ALERT_OP_CENTER: CHAR(32)
TOLL_RATE_NOTIFS_ENABLED_IND: NUMBER(1)
TOLL_RATE_NOTIF_RECIPIENT: VARCHAR2(35)
TRAVEL ROUTE DEST
*TR_ROUTE_ID: CHAR(32)
*SORT_ORDER_NUMBER: NUMBER(1)
ALT_DEST_TEXT: VARCHAR2(30)
EXTERNAL EVENT FILTER
*RULE_ID: CHAR(32)
RULE: VARCHAR2(4000)
ROUTE TOLL RATE
TR_ROUTE_ID: CHAR(32)
TOLL_RATE_EFF_TIME: DATE
TOLL_RATE_EXP_TIME: DATE
TOLL_RATE_CENTS: NUMBER(5)
TOLL_RATE_REASON_CODE: NUMBER(2)
TOLL_RATE_INAPPLICABLE_IND: NUMBER(1)
OBJECT LOCATION ALIAS
OBJECT_ID: CHAR(32)
PUBLIC_NAME: VARCHAR2(50)
INTERNAL_NAME: VARCHAR2(50)
DMS RELATED ROUTE
*DMS_DEVICE_ID: CHAR(32)
*TRAVEL_ROUTE_ID: CHAR(32)
ROUTE TOLL RATE TEXT
TR_ROUTE_ID: CHAR(32)
ROUTE_TOLL_RATE_EFF_TIME: DATE
ROUTE_TOLL_RATE_REASON_STR: VARCHAR2(1000)
TRAVEL ROUTE CONSUMER
*TR_ROUTE_ID: CHAR(32)
*SORT_ORDER_NUMBER: NUMBER(2)
CONSUMER_ID: CHAR(32)
PROXY_CONSUMER_ID: CHAR(32)
LINK TRAVEL TIME
RL_LINK_ID: CHAR(32)
LINK_TRAVEL_TIME_EFF_TIME: DATE
LINK_TRAVEL_TIME_SECS: NUMBER(5)
LINK_TRAVEL_TIME_QUAL: NUMBER(3)
LINK_TRAVEL_TIME_TREND: NUMBER(1)
OBJECT LOCATION
*OBJECT_ID: VARCHAR2(32)
LOCATION_TEXT: VARCHAR2(1024)
LOCATION_DESC_OVERRIDDEN: NUMBER(1)
COUNTY_NAME: VARCHAR2(50)
COUNTY_FIPS_CODE: CHAR(3)
COUNTY_CODE: NUMBER(3)
USPS_STATE_CODE: CHAR(2)
STATE_FULL_NAME: VARCHAR2(32)
STATE_FIPS_CODE: CHAR(2)
REGION_NAME: VARCHAR2(32)
ROUTE_SPEC_TYPE: NUMBER(1)
ROUTE_FREE_FORM_TEXT: VARCHAR2(40)
ROUTE_TYPE: NUMBER(1)
ROUTE_PREFIX: VARCHAR2(10)
ROUTE_NUMBER: VARCHAR2(10)
ROUTE_SUFFIX: VARCHAR2(10)
INT_FEAT_TYPE: NUMBER(1)
INT_FEAT_PROX_TYPE: NUMBER(2)
INT_FEAT_PROX_DIST: NUMBER(6)
ROAD_NAME: VARCHAR2(50)
INT_ROUTE_SPEC_TYPE: NUMBER(1)
INT_ROUTE_FREE_FORM_TEXT: VARCHAR2(40)
INT_ROUTE_TYPE: NUMBER(1)
INT_ROAD_NAME: VARCHAR2(50)
INT_ROUTE_PREFIX: VARCHAR2(10)
INT_ROUTE_NUMBER: VARCHAR2(10)
INT_ROUTE_SUFFIX: VARCHAR2(4)
INT_FEAT_MILEPOST_TYPE: NUMBER(1)
INT_FEAT_MILLI_MILEPOST_DATA: NUMBER(6)
ROADWAY_LOC_ALIAS_PUB_NAME: VARCHAR2(90)
ROADWAY_LOC_ALIAS_INT_NAME: VARCHAR2(90)
LATITUDE_UDEG: NUMBER(10)
LONGITUDE_UDEG: NUMBER(10)
GEOLOC_SOURCE_TYPE: NUMBER(2)
GEOLOC_SOURCE_DESC: VARCHAR2(35)
SHOW_ROUTE_NAME: NUMBER(1)
SHOW_INT_ROUTE_NAME: NUMBER(1)
DIRECTION_CODE: NUMBER(3)
OBJECT_TYPE: NUMBER(3)
TOLL RAW DATA
*TOLL_DATA_IMPORT_ID: NUMBER
*EXT_SYS_START_ID: VARCHAR2(35)
*EXT_SYS_END_ID: VARCHAR2(35)
EXT_SYS_ROUTE_DESC: VARCHAR2(127)
TOLL_RATE_EFF_TIME: DATE
TOLL_RATE_EXP_TIME: DATE
TOLL_RATE_CENTS: NUMBER(5)
LINK RAW DATA
*LINK_DATA_IMPORT_ID: NUMBER
*EXT_LINK_ID: CHAR(9)
LINK_TRAVEL_TIME_EFF_TIME: DATE
LINK_TRAVEL_TIME_SECS: NUMBER(5)
LINK_TRAVEL_TIME_QUAL: NUMBER(2)
LINK_SPEED_MPH: NUMBER(3)
DMS TRAV ROUTE MSG CONFIG LOG
*SYSTEM_TIMESTAMP: DATE
*DMS_DEVICE_ID: VARCHAR2(32)
DEVICE_NAME: VARCHAR2(15)
SCHEDULE_CONFIG_FLAG: NUMBER
LINK DATA IMPORT
*IMPORT_ID: NUMBER
SYSTEM_TIMESTAMP: DATE
EXT_SYS_NAME: VARCHAR2(35)
DMS TRAV ROUTE MSG
*MSG_ID: CHAR(32)
TEMPLATE_ID: CHAR(32)
AUTO_ROW_POSITIONING_IND: NUMBER(1)
TOLL DATA IMPORT
*IMPORT_ID: NUMBER
SYSTEM_TIMESTAMP: DATE
EXT_SYS_NAME: VARCHAR2(35)
LINK SMOOTHED DATA
*LINK_DATA_IMPORT_ID: NUMBER
*EXT_LINK_ID: CHAR(9)
LINK_TRAVEL_TIME_EFF_TIME: DATE
LINK_TRAVEL_TIME_SECS: NUMBER(5)
LINK_TRAVEL_TIME_QUAL: NUMBER(2)
LINK_SPEED_MPH: NUMBER(3)
EXTERNAL OBJECT EXCLUSION
*EXCLUSION_ID: CHAR(32)
EXTERNAL_OBJECT_ID: CHAR(256)
EXTERNAL_OBJECT_TYPE: NUMBER(3)
EXTERNAL_SYSTEM: VARCHAR2(35)
EXTERNAL_AGENCY: VARCHAR2(35)
ROADWAY LOCATION ALIAS
PUBLIC_NAME: VARCHAR2(50)
INTERNAL_NAME: VARCHAR2(50)
USPS_STATE_CODE: CHAR(2)
STATE_FIPS_CODE: CHAR(2)
REGION_NAME: VARCHAR2(32)
COUNTY_NAME: VARCHAR2(50)
COUNTY_FIPS_CODE: CHAR(3)
COUNTY_CODE: VARCHAR2(3)
DIRECTION_CODE: NUMBER(3)
ROUTE_SPEC_TYPE: NUMBER(1)
ROUTE_FREE_FORM_TEXT: VARCHAR2(50)
ROUTE_TYPE: NUMBER(1)
ROAD_NAME: VARCHAR2(50)
ROUTE_PREFIX: VARCHAR2(10)
ROUTE_NUMBER: VARCHAR2(10)
ROUTE_SUFFIX: VARCHAR2(10)
INT_FEAT_TYPE: NUMBER(1)
INT_FEAT_PROX_TYPE: NUMBER(2)
INT_FEAT_PROX_DIST: NUMBER(6)
INT_ROUTE_SPEC_TYPE: NUMBER(1)
INT_ROUTE_FREE_FORM_TEXT: VARCHAR2(50)
INT_ROUTE_TYPE: NUMBER(1)
INT_ROAD_NAME: VARCHAR2(50)
INT_ROUTE_PREFIX: VARCHAR2(10)
INT_ROUTE_NUMBER: VARCHAR2(10)
INT_ROUTE_SUFFIX: VARCHAR2(4)
INT_FEAT_MILEPOST_TYPE: NUMBER(1)
INT_FEAT_MILLI_MILEPOST_DATA: NUMBER(6)
DESCRIPTION: VARCHAR2(512)
STATE_FULL_NAME: VARCHAR2(32)
LATITUDE_UDEG: NUMBER(10)
LONGITUDE_UDEG: NUMBER(10)
DMS TRAV ROUTE MSG ROUTE
*DTRM_MSG_ID: CHAR(32)
*TRAVEL_ROUTE_ID: CHAR(32)
*SORT_ORDER_NUM: NUMBER(2)
ROADWAY LINK
*LINK_ID: CHAR(32)
EXT_SYS_NAME: VARCHAR2(10)
EXT_LINK_ID: VARCHAR2(9)
LINK_NAME: VARCHAR2(100)
USPS_STATE_CODE: CHAR(2)
STATE_FIPS_CODE: CHAR(2)
COUNTY_NAME: VARCHAR2(50)
COUNTY_FIPS_CODE: VARCHAR2(3)
ROUTE_SPEC_TYPE: NUMBER(1)
ROUTE_FREE_FORM_TEXT: VARCHAR2(50)
ROUTE_TYPE: NUMBER(1)
ROUTE_PREFIX: VARCHAR2(10)
ROUTE_NUMBER: VARCHAR2(100)
ROUTE_SUFFIX: VARCHAR2(10)
MILLI_MILES: NUMBER(5)
START_LAT_UDEG: NUMBER(9)
START_LONG_UDEG: NUMBER(10)
END_LAT_UDEG: NUMBER(9)
END_LONG_UDEG: NUMBER(10)
ROAD_NAME: VARCHAR2(100)
DIRECTION_CODE: NUMBER
DMS TRAV ROUTE MSG ROUTES LOG
*DMS_DEVICE_ID: CHAR(32)
*DMS_TRAV_ROUTE_MSG_ID: CHAR(32)
SYSTEM_TIMESTAMP: DATE
TRAV_ROUTE_ID: CHAR(32)
DMS TRAV ROUTE MSG ROUTE LOG
*SYSTEM_TIMESTAMP: DATE
*MSG_ROUTE_LOG_SEQUENCE: NUMBER
*DMS_DEVICE_ID: VARCHAR2(32)
DMS_TRAV_ROUTE_MSG_ID: VARCHAR2(32)
TR_ROUTE_ID: VARCHAR2(32)
DMS TRAV ROUTE MSG STATUS LOG
*SYSTEM_TIMESTAMP: DATE
*STAT_LOG_SEQUENCE: NUMBER
*DMS_DEVICE_ID: VARCHAR2(32)
DEVICE_NAME: VARCHAR2(15)
COMMUNICATION_MODE: NUMBER
OPERATIONAL_STATUS: NUMBER
SCHEDULE_ENABLED_INDICATOR: NUMBER
ENABLED_DMS_TRAV_ROUTE_MSG_ID: VARCHAR2(32)
DMS_MESSAGE: VARCHAR2(1024)
DMS_TRAV_ROUTE_MSG_STATE: NUMBER
DMS_TRAV_ROUTE_MSG_REASON: VARCHAR2(4000)
ROADWAY LOCATION STATE
*USPS_CODE: CHAR(2)
*FIPS_CODE: CHAR(2)
NAME: VARCHAR2(20)
DESCRIPTION: VARCHAR2(1024)
DMS TRAV TIME SCHEDULE
*DMS_DEVICE_ID: CHAR(32)
*START_HOUR: NUMBER(2)
*START_MIN: NUMBER(2)
*END_HOUR: NUMBER(2)
*END_MIN: NUMBER(2)
DMS TRAV ROUTE MSG MSGS LOG
*SYSTEM_TIMESTAMP: DATE
*MSGS_LOG_SEQUENCE: NUMBER
*DMS_DEVICE_ID: VARCHAR2(32)
DMS_TRAV_ROUTE_MSG_ID: VARCHAR2(32)
DMS_TRAV_ROUTE_MSG_TEMPLATE_ID: VARCHAR2(4000)
ROADWAY LOCATION REGION
*NAME: VARCHAR2(32)
USPS_STATE_CODE: CHAR(2)
DESCRIPTION: VARCHAR2(1024)
TRAVEL ROUTE LOCATION
*TR_ROUTE_ID: VARCHAR2(32)
*SORT_ORDER_NUMBER: NUMBER(2)
COUNTY_NAME: VARCHAR2(50)
COUNTY_FIPS_CODE: CHAR(3)
COUNTY_CODE: NUMBER(3)
DIRECTION_CODE: NUMBER(3)
USPS_STATE_CODE: CHAR(2)
STATE_FULL_NAME: VARCHAR2(32)
STATE_FIPS_CODE: CHAR(2)
ROUTE_SPEC_TYPE: NUMBER(1)
ROUTE_FREE_FORM_TEXT: VARCHAR2(40)
ROUTE_TYPE: NUMBER(1)
ROUTE_PREFIX: VARCHAR2(10)
ROUTE_NUMBER: VARCHAR2(10)
ROUTE_SUFFIX: VARCHAR2(10)
ROAD_NAME: VARCHAR2(50)
DB_CODE: CHAR(1)
TOLL RATE ROUTES
*TOLL_RATE_EXT_SYS_NAME: VARCHAR2(35)
*TOLL_RATE_EXT_START_ID: VARCHAR2(35)
*TOLL_RATE_EXT_END_ID: VARCHAR2(35)
TOLL_RATE_EXT_DESC: VARCHAR2(127)
LAST_RECEIVED_TIME: DATE
ROUTE TRAVEL TIME TEXT
*TR_ROUTE_ID: CHAR(32)
*ROUTE_TRAVEL_TIME_EFF_TIME: DATE
ROUTE_TRAVEL_TIME_CALC: VARCHAR2(1000)
ROUTE_TRAVEL_TIME_REASON_CODE: NUMBER(2)
TRAVEL ROUTE LINK
*TR_ROUTE_ID: CHAR(32)
*SORT_ORDER_NUMBER: NUMBER(4)
RL_LINK_ID: CHAR(32)
PERCENT: NUMBER(3)
MIN_ALLOWED_QUALITY: NUMBER(3)
ROUTE TRAVEL TIME
TR_ROUTE_ID: CHAR(32)
ROUTE_TRAVEL_TIME_EFF_TIME: DATE
ROUTE_TRAVEL_TIME_SECS: NUMBER(5)
ROUTE_TRAVEL_TIME_TREND: NUMBER(1)
TRAVEL_TIME_INAPPLICABLE_IND: NUMBER(1)
ROUTE_ACT_TRAVEL_TIME_SECS: NUMBER(5)
Travel Route and Toll Rate*
Pg13
1, 1 / 1, 2 -- 4:45:19 PM , 5/11/2011

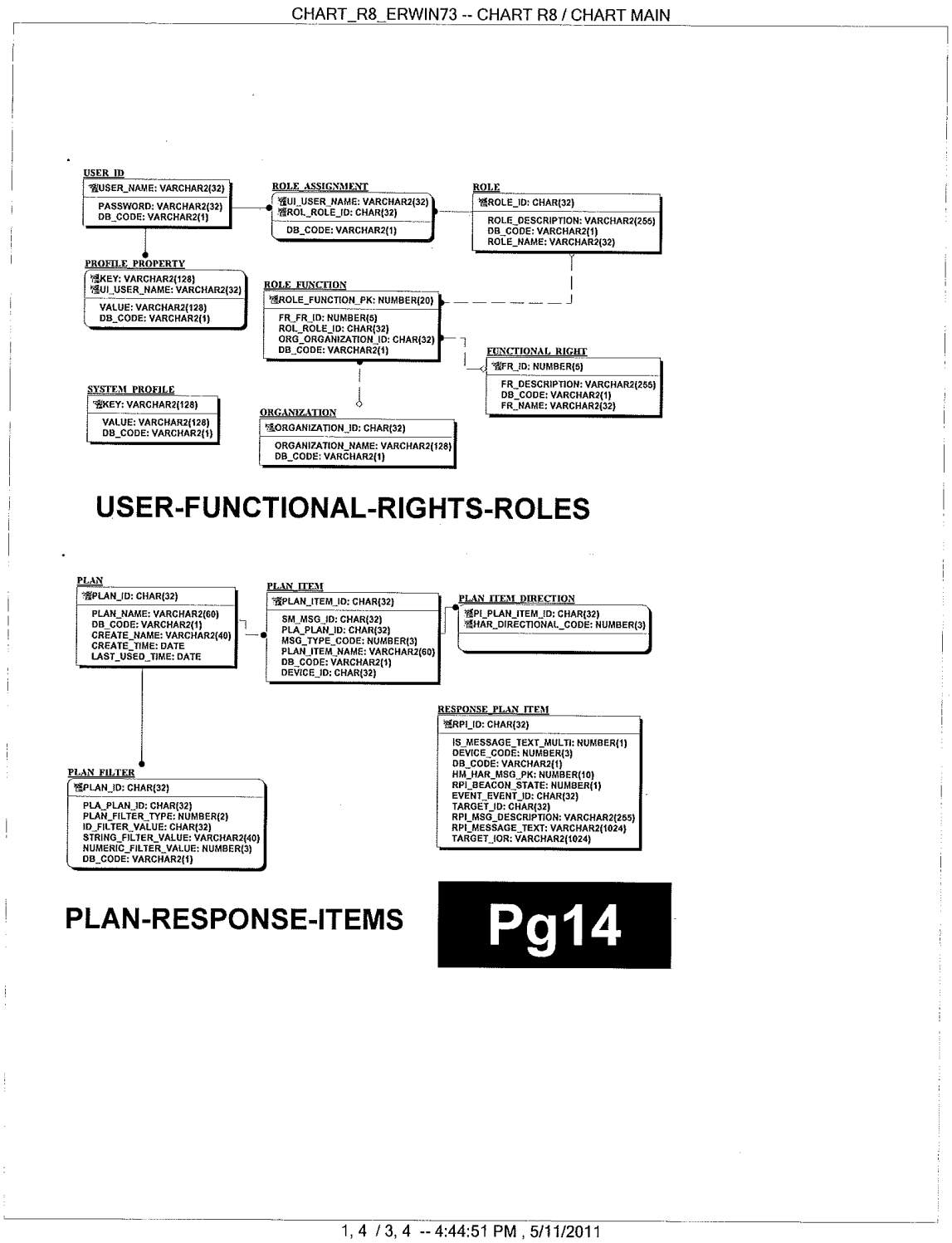


Figure 2-4 R8 ERD

2.4.1.1.2.2 Function to Entity Matrix Report

The Create, Retrieve, Update, Delete (CRUD) matrix cross-references business functions to entities and shows the use of the entities by those functions. This report will be generated as part of the CHART O&M Guide.

2.4.1.1.2.3 Table Definition Report –

In existing tables shown below:

- Deleted columns/constraints marked with a minus sign ("-")
- Modified columns/constraints marked with an asterisk ("*")
- New columns/constraints marked with a plus sign ("+")

2.4.1.1.2.3.1 Tables Modified for the IP HAR / IP SHAZAM features

2.4.1.1.2.3.1.1 CHART

HAR Table

The HAR table is modified to make fields specific to Telephony communications allow NULL and to add new columns for TCP/IP communications and for configuration values specific to the DR1500 HAR when operated via TCP/IP.

```
CREATE TABLE HAR
 (
     DEVICE_ID CHAR(32) NOT NULL,
     HAR_MODEL_ID NUMBER(5) NOT NULL,
     ORG_ORGANIZATION_ID CHAR(32) NOT NULL,
     DB_CODE VARCHAR2(1),
     DEVICE_NAME VARCHAR2(15) NOT NULL,
     DEVICE_LOCATION VARCHAR2(60),
*    HAR_ACCESS_PIN VARCHAR2(7),
*    DEFAULT_PHONE_NUMBER VARCHAR2(25),
     DEFAULT_MONITOR_PHONE_NUMBER VARCHAR2(25),
     MAX_TIME NUMBER(5),
     PORT_TYPE NUMBER(1) NOT NULL,
*    PORT_MANAGER_TIMEOUT NUMBER(10),
     MONITOR_PORT_TYPE NUMBER(1),
     MONITOR_PORT_MANAGER_TIMEOUT NUMBER(10),
     DEFAULT_HEADER_CLIP_PK NUMBER(20) NOT NULL,
     DEFAULT_BODY_CLIP_PK NUMBER(20) NOT NULL,
     DEFAULT_TRAILER_CLIP_PK NUMBER(20) NOT NULL,
     CREATED_TIMESTAMP DATE,
     UPDATED_TIMESTAMP DATE,
     ENABLE_DEVICE_LOG NUMBER(1) DEFAULT 0 NOT NULL,
     MASTER_HAR_ID CHAR(32) DEFAULT '00000000000000000000000000000000'
        NOT NULL,
     BAND CHAR(2) NOT NULL,
     CALL_SIGN VARCHAR2(64),
```

```
     FREQUENCY_KHZ NUMBER(6) NOT NULL,
     MAINT_ORGANIZATION_ID CHAR(32),
     DISABLE_DTMF_RESPONSE_MODE NUMBER(1) DEFAULT 0 NOT NULL,
+    TCP_HOST VARCHAR2(16),
+    TCP_PORT NUMBER(5),
+    MIN_DC_VOLTAGE NUMBER(3),
+    MAX_VSWR NUMBER(3),
     CONSTRAINT HAR_PK PRIMARY KEY (DEVICE_ID),
     CONSTRAINT HAR_NAME_UK UNIQUE (DEVICE_NAME),
     CONSTRAINT HAR_PORT_TYPE_CK CHECK (port_type BETWEEN 0 AND 3),
     CONSTRAINT HAR_MONITOR_PORT_MNGR_T_0_CK CHECK
        (monitor_port_manager_timeout BETWEEN 0 AND 2147483647),
     CONSTRAINT HAR_MONITOR_PORT_TYPE_CK CHECK
        (monitor_port_type BETWEEN 0 AND 3),
     CONSTRAINT HAR_MODEL_ID_CK CHECK (har_model_id BETWEEN 0 AND 65535),
     CONSTRAINT HAR_PORT_MANAGER_TIMEOUT_CK CHECK
        (port_manager_timeout BETWEEN 0 AND 2147483647)
    );
```

HAR Status Table

The HAR Status table is modified to include the last time the setup command was run on the HAR and status values specific to the DR1500 HAR model that are available when communicating to the HAR using TCP/IP.

```
 CREATE TABLE HAR_STATUS
 (
     HAR_DEVICE_ID CHAR(32) NOT NULL,
     CEN_CENTER_ID CHAR(32),
     DEVICE_STATE_CODE NUMBER(3) NOT NULL,
     TRANSMITTER_STATE NUMBER(1) NOT NULL,
     HAR_INITIALIZED NUMBER(1) NOT NULL,
     COMM_STATUS NUMBER(1) NOT NULL,
     LAST_CONTACT_TIME DATE,
     STATUS_CHANGE_TIME DATE,
     LAST_DATESTAMP_REFRESH_TIME DATE,
     HM_HAR_MSG_PK NUMBER(10) NOT NULL,
+    LAST_SETUP_TIME DATE,
+    LAST_STATUS_MISMATCH_TIME DATE,
+    POWER_STATUS NUMBER(1),
+    DC_VOLTAGE NUMBER(3),
+    BROADCAST_MONITOR_PCT NUMBER(3),
+    HAR_MODE NUMBER(1),
+    HAR_SUB_MODE NUMBER(1),
+    HAR_SYNC_MODE NUMBER(1),
+    XMIT_SET_POWER NUMBER(3),
+    XMIT_FORWARD_POWER NUMBER(3),
+    XMIT_REFLECTED_POWER NUMBER(3),
+    XMIT_VSWR NUMBER(3),
+    XMIT_MODULATION_PCT NUMBER(3),
+    DCC_VERSION_INFO VARCHAR2(128),
+    HAR_VERSION_INFO VARCHAR2(128),
+    HAR_TIMESTAMP DATE,
```

```
     CONSTRAINT HS_PK PRIMARY KEY (HAR_DEVICE_ID),
     CONSTRAINT HS_HAR_FK FOREIGN KEY (HAR_DEVICE_ID)
        REFERENCES HAR (DEVICE_ID) ON DELETE
        CASCADE NOT DEFERRABLE INITIALLY IMMEDIATE VALIDATE,
     CONSTRAINT HS_HM_FK FOREIGN KEY (HM_HAR_MSG_PK)
        REFERENCES HAR_MSG (HAR_MSG_PK),
     CONSTRAINT HAR_INITIALIZED_CK CHECK (har_initialized IN (0,1)),
     CONSTRAINT HAR_DEVICE_STATE_CODE_CK CHECK
        (device_state_code BETWEEN 0 AND 2),
     CONSTRAINT TRANSMITTER_STATE_CK CHECK (transmitter_state IN (0,1)),
     CONSTRAINT HAR_COMM_STATUS_CK CHECK (COMM_STATUS BETWEEN 0 AND 3)
 );
```

SHAZAM Table

The SHAZAM table is modified to change the Telephony specific columns to accept NULL and to add columns needed for TCP/IP communications and the Relay Number needed by the HWG-ER02a SHAZAM model.

```
 CREATE TABLE SHAZAM
 (
     DEVICE_ID CHAR(32) NOT NULL,
     SHAZAM_MODEL_ID NUMBER(5) NOT NULL,
     ORG_ORGANIZATION_ID CHAR(32) NOT NULL,
     DB_CODE VARCHAR2(1),
     DEVICE_NAME VARCHAR2(15) NOT NULL,
     DEVICE_LOCATION VARCHAR2(128),
     HAR_DEVICE_ID CHAR(32),
     SHAZAM_ACCESS_PIN VARCHAR2(3),
*    DEFAULT_PHONE_NUMBER VARCHAR2(25),
     SHAZAM_DIRECTIONAL_CODE NUMBER(3),
     REFRESH_INTERVAL NUMBER(5),
     REFRESH_ENABLED NUMBER(1) NOT NULL,
     PORT_TYPE NUMBER(1) NOT NULL,
*    PORT_MANAGER_TIMEOUT NUMBER(10),
     CREATED_TIMESTAMP DATE,
     UPDATED_TIMESTAMP DATE,
     MESSAGE VARCHAR2(256),
     MAINT_ORGANIZATION_ID CHAR(32),
+    TCP_HOST VARCHAR2(16),
+    TCP_PORT NUMBER(5),
+    RELAY_NUMBER NUMBER(2),
     CONSTRAINT SHAZAM_PK PRIMARY KEY (DEVICE_ID),
     CONSTRAINT SHAZAM_NAME_UK UNIQUE (DEVICE_NAME),
     CONSTRAINT SHAZAM_DIRECTIONAL_CODE_CK CHECK
        (SHAZAM_DIRECTIONAL_CODE BETWEEN 0 AND 255),
     CONSTRAINT REFRESH_INTERVAL_CK CHECK
        (refresh_interval BETWEEN 0 AND 65535),
     CONSTRAINT SHAZAM_MODEL_ID_CK CHECK
        (shazam_model_id BETWEEN 0 AND 65535),
     CONSTRAINT SHAZAM_PORT_MANAGER_TIMEOUT_CK CHECK
        (port_manager_timeout BETWEEN 0 AND 2147483647),
     CONSTRAINT SHAZAM_PORT_TYPE_CK CHECK (port_type BETWEEN 0 AND 3)
```

```
 );
```

SHAZAM_STATUS Table

The SHAZAM_STATUS table is modified to add a column to store the actual beacon status as queried from the device. This column only applies to the HWG-ER02a model and will be NULL for Viking RC2A SHAZAMs.

```
CREATE TABLE SHAZAM_STATUS
(
    SHAZAM_DEVICE_ID CHAR(32) NOT NULL,
    CEN_CENTER_ID CHAR(32),
    DEVICE_STATE_CODE NUMBER(3) NOT NULL,
    BEACON_STATE NUMBER(1),
    COMM_STATUS NUMBER(1) NOT NULL,
    STATUS_CHANGE_TIME DATE,
    LAST_ATTEMPTED_REFRESH_TIME DATE,
    LAST_CONTACT_TIME DATE,
+   BEACON_STATE_ACTUAL NUMBER(1),
    CONSTRAINT SS_PK PRIMARY KEY (SHAZAM_DEVICE_ID),
    CONSTRAINT SS_SHAZAM_FK FOREIGN KEY (SHAZAM_DEVICE_ID)
       REFERENCES SHAZAM (DEVICE_ID)
       ON DELETE CASCADE NOT DEFERRABLE INITIALLY IMMEDIATE VALIDATE,
    CONSTRAINT SHAZAM_COMM_STATUS_CK CHECK (comm_status BETWEEN 0 AND 2),
    CONSTRAINT SHAZAM_DEVICE_STATE_CODE_CK CHECK
       (device_state_code BETWEEN 0 AND 2),
    CONSTRAINT SHAZAM_BEACON_STATE_CK CHECK (beacon_state IN (0,1))
);
```

DEVICE_ALERT_NOTIFICATION Table (NEW)

This is a new table that is used to store alert and notification settings for a device. For R8, this will be used for both HAR and SHAZAM devices, and is available for use for other device types in the future. There will be one row in this table for each SHAZAM in the SHAZAM table, and one row in this table for each HAR in the HAR table.

```
CREATE TABLE DEVICE_ALERT_NOTIFICATION
 (
     OBJECT_ID VARCHAR2(32),
     DEVICE_TYPE   NUMBER(3),
     COMMFAIL_ALERT_CENTER_ID CHAR(32),
     HWFAIL_ALERT_CENTER_ID CHAR(32),
     COMMFAIL_NOTIF_GROUP_ID   NUMBER(5),
     COMMFAIL_NOTIF_GROUP_NAME VARCHAR2(50),
     HWFAIL_NOTIF_GROUP_ID NUMBER(5),
     HWFAIL_NOTIF_GROUP_NAME VARCHAR2(50),
     CONSTRAINT PK_DEVICE_ALERT_NOTIFICATION PRIMARY KEY (OBJECT_ID)
 );
```

The following scripts will be used to add rows to DEVICE_ALERT_NOTIFICATION for HARs and SHAZAMs that already exist in the database at the time of R8 deployment:

```
INSERT INTO DEVICE_ALERT_NOTIFICATION
(
  OBJECT_ID,
  DEVICE_TYPE,
  COMMFAIL_ALERT_CENTER_ID,
  HWFAIL_ALERT_CENTER_ID,
  COMMFAIL_NOTIF_GROUP_ID,
  COMMFAIL_NOTIF_GROUP_NAME,
  HWFAIL_NOTIF_GROUP_ID,
  HWFAIL_NOTIF_GROUP_NAME
)
SELECT DEVICE_ID, 6, '00000000000000000000000000000000',
        '00000000000000000000000000000000',
        NULL,NULL,NULL,NULL
FROM HAR;
```

```
INSERT INTO DEVICE_ALERT_NOTIFICATION(
     OBJECT_ID,
     DEVICE_TYPE,
     COMMFAIL_ALERT_CENTER_ID,
     HWFAIL_ALERT_CENTER_ID,
     COMMFAIL_NOTIF_GROUP_ID,
     COMMFAIL_NOTIF_GROUP_NAME,
     HWFAIL_NOTIF_GROUP_ID,
     HWFAIL_NOTIF_GROUP_NAME)
SELECT DEVICE_ID, 255, '00000000000000000000000000000000',
        '00000000000000000000000000000000',
        NULL,NULL,NULL,NULL
FROM SHAZAM;
```

2.4.1.1.2.3.1.2 Mapping

There are no mapping related database changes for CHART R8.

2.4.1.1.2.4 PL/SQL Module Definition and Database Trigger Reports

There are no new PL/SQL modules for CHART R8.

2.4.1.1.2.5 Database Size Estimate - provides size estimate of current design

There are no changes of any significance to the database size for R8.

2.4.1.1.2.6 Data Distribution

There are no changes to data distribution for R8.

2.4.1.1.2.7 Database Replication

There are no changes to database replication for R8.

2.4.1.1.2.8 Archival Migration

There are no changes to archival migration for R8.

2.4.1.1.2.9 Database Failover Strategy

There are no changes to the database failover strategy for R8.

2.4.1.1.2.10 Reports

No reports will be added or updated for R8. Since R5, the CHART reporting function has been transferred to University of Maryland.

2.4.1.2 CHART Flat Files

The following describes the use of flat files in CHART.

2.4.1.2.1 Service Registration Files

There are no new Java services and therefore no new service registration files for CHART R8.

2.4.1.2.2 Service Property Files

There are only minor changes to existing service property files for CHART R8. Communication timeout parameters are added for both HAR and SHAZAM devices in the HAR Service property file.

2.4.1.2.3 GUI Property Files

There are no updates to the GUI properties file in its WEB-INF directory for CHART R8.

2.4.1.2.4 Arbitration Queue Storage Files

There are no changes to Arbitration Queue Storage Files for R8.

2.4.1.2.5 Device Logs

There are no changes to Device Log Files for R8.

2.4.1.2.6 Traffic Sensor Raw Data Logs

There are no changes to Traffic Sensor Raw Data Log Files for R8.

2.4.1.2.7 Service Process Logs

All CHART services write to a process log, used to provide a historical record of activity undertaken by the services. These logs are occasionally referenced by software engineering

personnel to diagnose a problem or reconstruct a sequence of events leading to a particular anomalous situation. These logs are automatically deleted by the system after a set period of time defined by the service's properties file, so they do not accumulate infinitely. These files are stored in the individual service directories and are named by the service name and date, plus a "`.txt`" extension. These logs are typically read only by software engineering personnel. Except where noted, there are no changes for service process logs for R8 features.

2.4.1.2.8 Service Error Logs

All CHART services write to an error log, used to provide detail on certain errors encountered by the services. Most messages, including most errors, are captured by the CHART software and written to the process logs, but certain messages (typically produced by the Java Virtual Machine itself, by COTS, or DLLs) cannot be captured by CHART Software and instead are captured in these "catch-all" logs. Errors stored in these logs are typically problems resulting from a bad installation; once the system is up and running, errors rarely appear in these error logs. Debugging information from the JacORB COTS, which is not usually indicative of errors, can routinely be found in these error logs, as well. These log files can be reviewed by software engineering personnel to diagnose an installation problem or other type of problem. These logs are automatically deleted by the system after a set period of time defined by the service's properties file, so they do not accumulate infinitely. These files are stored in the individual service directories and are named by the service name and date, plus an ".err" extension. These logs are typically read only by software engineering personnel. Except where noted, there are no changes for service error logs for R8 features.

2.4.1.2.9 GUI Process Logs

Like the CHART background services, the CHART GUI service also writes to a process log file, used to provide a historical record of activity undertaken by the process. These GUI process logs are occasionally referenced by software engineering personnel to diagnose a problem or reconstruct a sequence of events leading to a particular anomalous situation. These logs are automatically deleted by the system after a set period of time defined by the GUI service's properties file, so they do not accumulate infinitely. These files are stored in the `chartlite/LogFiles/` directory under the `WebApps/` directory in the Apache Tomcat installation area. They are named by the service name ("`chartlite`") and date, plus a "`.txt`" extension. These logs are typically read only by software engineering personnel. Additional log files written by the Apache Tomcat system itself are stored in the `log/` directory in the Apache Tomcat installation area.

- R8 GUI changes do not change the way the GUI process logs operate.

2.4.1.2.10 FMS Port Configuration Files

The CHART Communications Services read a Port Configuration file, typically named `PortConfig.xml`, upon startup, which indicates which ports are to be used by the service and how they are to be initialized. A Port Configuration Utility is provided which allows for addition, removal of ports and editing of initialization parameters. As indicated by the extension, these files are in XML format. This means these files are hand-editable, although the Port

Configuration Utility allows for safer, more controlled editing. The Port Configuration files are typically modified only by software engineers or telecommunications engineers.

- There are no changes to this section for the any of the R8 features.

2.4.1.2.11 Watchdog Configuration Files

There are no changes to the watchdog configuration files for any of the R8 features.

2.4.2 Database Design

Changes made to the CHART database design for Release 8 features are described below.

2.4.2.1 HAR and SHAZAM

2.4.2.1.1 CHART

The changes to the database design for R8 are detailed above in section 2.4.1. Following is a description of these changes:

HAR Table

The HAR Table is updated to allow NULL values in columns that are specific to Telephony communications, for those columns will be empty when a HAR is set up to use TCP/IP communications. New columns are added to support TCP/IP communications. New columns are also added for thresholds that are set for use in detecting hardware failures in DR1500 HARs that are set to use TCP/IP communications. (They are not used for DR1500 HARs that use Telephony based communications because the system cannot query the status of the HAR when using Telephony).

HAR_STATUS Table

The HAR_STATUS table is updated to include new columns for status values that can be retreieved from DR1500 HARs when TCP/IP communications are used.

SHAZAM Table

The SHAZAM table is updated to allow NULL in columns specific to Telephony communications and to add columns used for TCP/IP communications. A column is also added to identify the relay that is to be used on the HWG-ER02a SHAZAM. This model SHAZAM contains 2 relays, only one of which is used in CHART (and by the device to cause the beacons to flash on the SHAZAM sign.)

SHAZAM_STATUS Table

The SHAZAM_STATUS table is updated to include a new column that is used to store the actual SHAZAM beacon state as read from the SHAZAM. This is only supported for the HWG-ER02a model and will always be NULL for the Viking RC2A model SHAZAM.

DEVICE_ALERT_NOTIFICATION Table

The DEVICE_ALERT_NOTIFICATION table is new for R8. It is used to store alert and notification settings for devices. In R8, this table is used for HAR and SHAZAM devices, but is available for use by other devices in future releases. There will be one row in this table for each HAR and SHAZAM device defined in the HAR and SHAZAM tables. Use of a common table allows us to keep these fields out of the HAR and SHAZAM tables and allows for code reuse.

2.4.2.1.2 Intranet Mapping

There are no Intranet Mapping database design changes for R8.

3 Key Design Concepts

3.1 IP HAR

Background

The IP HAR feature includes changes to support HARs that are connected to CHART via the TCP/IP network. CHART already supports DR1500 HARs that are connected via POTS and use of a Telephony port. An optional module known as the Digital Communications Controller (DCC) can be added to a DR1500 HAR to enable TCP/IP communications. Several existing DR1500 HARs are being outfitted with this module, and other new HARs that already contain this module are being added to the system. There are other DR1500 HARs that are currently fielded that do not have a DCC and will not have a DCC added. CHART will continue to communicate with those DR1500 HARs using POTS and a Telephony port.

Communications

When adding a HAR to the system, administrators will now be able to select TCP/IP communications for the DR1500 model. When TCP/IP communications are selected for a DR1500 HAR, the user may also enable polling of the device to have the system periodically check the status of the HAR. The communications type can be edited after a HAR is initially added to the system using the Control Line Communications Settings form.

Polling

When a DR1500 HAR is set to use TCP/IP communications and polling is enabled, the system will query its status on the interval as specified in the configuration. During each status poll, the system will check the HAR state as reported by the device against the HAR state as specified in CHART to determine if the HAR indicates it is doing what CHART last commanded it to do. CHART will check the play list and the transmitter on/off status to determine if there is a status match. If the status does not match, CHART will automatically queue a setup command for the HAR to restore clips to the HAR. The setup command downloads clips, sets the play list, and sets the transmitter on/off according the state specified in CHART.

Another function perfomed during a poll of a DR1500 is to check status values against configured thresholds for those values. Configuration values are included to allow thresholds for various status values to be specified, and these values will be used by the system during polling to determine if a hardware failure condition exists. If during a status poll CHART determines a status value lies outside the configured threshold it will set the HAR status to hardware failed. The values CHART will check against thesholds are the DC Voltage and Voltage Standing Wave Ratio (VSWR).

The status values obtained from the most recent poll of the HAR are displayed on the HAR's details page within the GUI and will indicate which values (if any) are found to be outside the configured thresholds and therefore cause a hardware failure condition.

Monitoring HAR Audio
When a DR1500 HAR is operated using TCP/IP communications, monitoring the HAR's audio is not possible. If a POTS line is connected to the HAR operators can dial into a DR1500 HAR and issue commands to determine its current message as they do prior to R8 support for TCP/IP communications.

Alerts and Notifications
CHART will allow alert and notification settings to be set for each HAR, regardless of its model and the type of communications used to control it. Separate values are supported to specify the op center to receive communication failure alerts, the op center to receive hardware failure alerts, the notification group to recive notifications of communication failures, and the notification group to receive notifications of hardware failures. Any or all of these values can be set to "None" to disable that particular alert or notification. When enabled, if CHART detects a status change related to the given type of failure (hardware or communication), CHART will create an alert and assign it to the specified operations center and/or will send a notification to the specified notification group.

Some HAR models (such as the AP55 and DR1500 configured to use a Telephony port) do not support polling or retrieving status, so these HARs will never raise a hardware failed condition and therefore setting an op center and/or notification group for those HARs will serve little purpose. The alert and notification feature is being developed generically to apply to all HARs to avoid rework in the future if support for new HAR models is added to the system.

3.2 IP SHAZAM

Background
A new SHAZAM model is being introduced in the systsem. This SHAZAM is known as the HWG-ER02a and is a simple device that provides network access to two electronic relays. One of the relays will be connected to the beacon ciruitry of the highway sign the SHAZAM is attached to, and the other relay will be unused. When a SHAZAM is activated in CHART, CHART will close the relay that is used to turn the beacons on. Likewise, when the SHAZAM is deactivated in CHART, CHART will open the relay which will cause the beacons to turn off.

Communications
When adding a SHAZAM to the system, administrators will now be able to select the model of the SHAZAM being added. The existing model, Viking RC2A will exist as a choice in addition to the new model, HWG-ER02a. When a Viking RC2A model is selected, Telephony port communications will automatically be selected and the administrator can enter the related fields as they do today, including the default phone number, access code, port manager connection timeout, and the port manager configuration. When the ER02a model is selected, TCP/IP communications will automatically be selected and the administrator can enter the IP address and port.

Refresh / Polling
The CHART system currently provides a refresh feature for SHAZAM devices. This feature allows refresh to be enabled for a SHAZAM and for a refresh interval to be specified. When refresh is enabled, CHART periodically connects to the SHAZAM and issues commands to set the SHAZAMs beacons to the state currently specified in CHART (beacons on or off). This feature will also be supported for the new HWG-ER02a model being added in R8. The one difference is that the HWG-ER02a model allows the status of its relay (and thus the status of the beacons) to be queried. When doing a refresh for a HWG-ER02a SHAZAM, CHART will check the status of the relay after the refresh command to determine if the relay is set as CHART commanded it. If not, CHART will set the SHAZAMs status to indicate a hardware failure.

Alerts and Notifications
CHART will allow alert and notification settings to be set for each SHAZAM, regardless of its model and the type of communications used to control it. Separate values are supported to specify the op center to receive communication failure alerts, the op center to receive hardware failure alerts, the notification group to recive notifications of communication failures, and the notification group to receive notifications of hardware failures. Any or all of these values can be set to "None" to disable that particular alert or notification. When enabled, if CHART detects a status change related to the given type of failure (hardware or communication), CHART will create an alert and assign it to the specified operations center and/or will send a notification to the specified notification group.

Only the HWG-ER02a SHAZAM model supports detecting a hardware failure. The Viking RC2A will never raise a hardware failed condition and therefore setting an op center and/or notification group for those SHAZAMs will serve little purpose. The alert and notification feature is being developed generically to apply to all SHAZAMs to avoid rework in the future if support for new SHAZAM models are added to the system.

3.3 Error Processing

In general, CHART traps conditions at both the GUI and at the server. User errors that are trapped by the GUI are reported immediately back to the user. The GUI will also report communications problems with the server back to the user. The server may also trap user errors and those messages will be written to a server log file and returned back to the GUI for display to the user. Additionally, server errors due to network errors or internal server problems will be written to log files and returned back to the GUI.

3.4 Packaging

3.4.1 CHART

This software design is broken into packages of related classes. The table below shows each package that is new or changed to support the Release 7 features.

Package Name	Package Description
CHART2.DeviceManagement	This CORBA package will be modified for R8 to define settings that

Package Name	Package Description
	can be used to control alerts and notifications for devices.
CHART2.DeviceUtility	This package is changed for R8 to add utility classes related to creating device alerts and notifications.
CHART2.HARControl	This CORBA package will be modified for R8 to add support for TCP/IP HAR communications and various configuration and status fields.
CHART2.HARControlModule	This package is changed for R8 to support the IP HAR and HAR device alerts and notifications.
CHART2.HARNotification	This CORBA package will be modified for R8 to add support for TCP/IP communications for SHAZAM devices and to add support for the HWG-ER02a SHAZAM model.
CHART2.SHAZAMControlModule	This package will be modified for R8 to add support for the HWG-ER02a SHAZAM and SHAZAM device alerts and notifications.
CHART2.Utility	This package will be changed in R8 to add some utiltiy methods to the ByteUtil package that will be used when communicating with the IP HAR.
chartlite.data	This GUI package will be changed in R8 to add generic support for device alerts and notification settings.
chartlite.data.har	This GUI package will be changed in R8 to support the new configuration and status settings of the IP HAR.
chartlite.data.shazam	This GUI package will been changed in R8 to support the new SHAZAM model.
chartlite.servlet.har	This GUI package will be modified in R8 to add support for TCP/IP HAR communications, polling, and alert and notification settings.
chartlite.servlet.shazam	This GUI package will be modified in R8 to add support for the new SHAZAM model.

3.4.2 Mapping

There are no mapping related packages (namespaces) added or modified for R8.

Namespace Name	Namespace Description
None	None

3.5 Assumptions and Constraints

1. The CHART software will be built to support the HAR DCC module with firmware version 5.26. All DR1500 HARs that are to be controlled via CHART R8 using TCP/IP communications must have firmware version 5.26 on their DCC.
2. CHART will continue to support DR1500 HARs that do not use a DCC and instead continue to use POTS and a Telphony port for control.

4 Use Cases – IP HAR and IP SHAZAM

This section includes CHART use case diagrams for the IP HAR and IP SHAZAM features. There are no changes to the Mapping application for R8 and therefore no use case diagrams for Mapping are included.

4.1 CHART

The use case diagrams depict new functionality for the CHART IP HAR and IP SHAZAM features. The use case diagrams exist in the Tau design tool in the Release8 area. The sections below indicate the title of the use case diagrams that apply to the IP HAR and IP SHAZAM.

4.1.1 Configure HAR (Use Case Diagram)

This diagram contains settings related to configuration of HAR and SHAZAM devices.

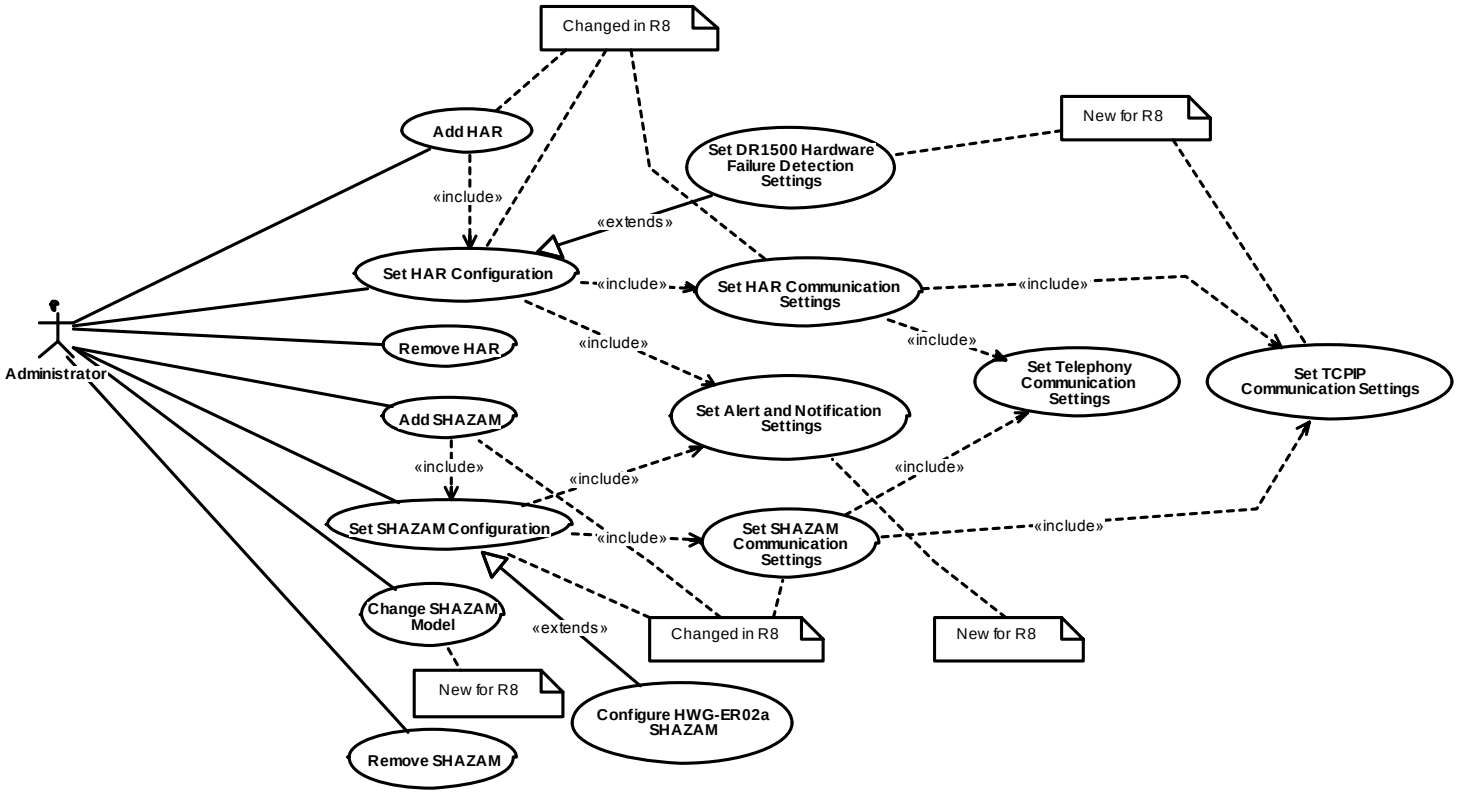


Figure 4-1. ConfigureHAR (Use Case Diagram)

4.1.1.1 Add HAR (Use Case)

A user with appropriate privileges may add a HAR to the system. The settings for the HAR must be specified when the HAR is added.

4.1.1.2 Add SHAZAM (Use Case)

A user with appropriate privileges can add a SHAZAM to the system. The settings for the SHAZAM must be specified at the

time the SHAZAM is added.

4.1.1.3 Administrator (Actor)

An administrator is a CHART user that has functional rights assigned to allow them to perform administrative tasks, such as system configuration and maintenance.

4.1.1.4 Change SHAZAM Model (Use Case)

The administrator can change the model of a SHAZAM if the SHAZAM is offline. When this occurs, any data specific to the target SHAZAM model will be defaulted and will require a separate edit to update (see Edit SHAZAM Configuration). This operation will keep the association between a HAR and the SHAZAM in tact (if any such association exists).

4.1.1.5 Configure HWG-ER02a SHAZAM (Use Case)

The system shall support setting the relay that is in use on a HWG-ER02a SHAZAM (Relay 1 or Relay 2).

4.1.1.6 Remove HAR (Use Case)

The system shall allow a suitably privileged user to remove an offline HAR from the system.

4.1.1.7 Remove SHAZAM (Use Case)

The system shall allow a suitably privileged user to remove an offline SHAZAM from the system.

4.1.1.8 Set Alert and Notification Settings (Use Case)

The system supports settings for a device used to specify which operations centers are to be notified of device failures and device communication failures. Similarly, settings exist to specify the notification group to recieve hardware failure notifications and the group to receive communication failure notifications. All 4 of these settings support a value of None to allow the associated alerts and notifications to be disabled.

4.1.1.9 Set DR1500 Hardware Failure Detection Settings (Use Case)

The system supports configuration settings for the DR1500 HAR that allow the system to detect when the HAR is hardware

failed. These settings are only applicable when the system is configured to use TCP/IP to communicate with the HAR. The settings include minimum DC voltage and the maximum voltage standing wave ratio (VSWR).

4.1.1.10 Set HAR Communication Settings (Use Case)

The system supports settings that determine how the system is to communicate with a HAR. Control line settings specify how to communicate with the HAR to control the device (set message etc.) while monitor line settings specify how to communicate with the HAR to monitor its current broadcast. The monitor line is only available for the AP55 HAR model, and only Telephony communications are supported for the monitor line. The control line settings can be Telephony or TCP/IP, however the AP55 model does not support TCP/IP communications. An access code is required when Telephony port communications are specified. The default TCP/IP port for HIS DR1500 HARs shall be 200.

4.1.1.11 Set HAR Configuration (Use Case)

The system shall allow a suitably privileged user to set the HAR configuration settings. This can be done as part of adding a HAR or on an existing HAR. The settings include the HAR name, location, and other basic settings. Default header, trailer, and message clips are included as part of the configuration. Communication settings and alert and notification settings are also included. Polling settings are included for the DR1500 HAR when TCP/IP communications are specified, including the ability to enable or disable polling and the polling rate (which defaults to 5 minutes).

4.1.1.12 Set SHAZAM Communication Settings (Use Case)

The system shall allow the communication settings for a SHAZAM to be set. The system supports both Telephony and TCP/IP communications. The Viking RC2A SHAZAM supports only Telephony communications and the HWG-ER02a SHAZAM supports only TCP/IP communications.

4.1.1.13 Set SHAZAM Configuration (Use Case)

The system allows the configuration settings for a SHAZAM to be specified while adding a SHAZAM to the system and allows the settings to be changed for SHAZAM devices that already exist in the system. The settings include the name, location, and other basic settings. Also included are communication settings and alert and notification settings. Automatic refresh can be enabled and the refresh interval can be specified. The default refresh interval for SHAZAMs that use TCP/IP communications is 5 minutes. The system supports two SHAZAM models, the Viking RC2A and the HWG-ER02a. The user can select the SHAZAM model when adding a SHAZAM. The model can be changed for an existing SHAZAM via a

separate operation (see Change SHAZAM Model). The access code for a Viking RC2A is included as part of its configuration, and the relay to be used on a HWG-ER02a is included as part of its configuration.

4.1.1.14 Set TCPIP Communication Settings (Use Case)

The system allows TCP/IP communication settings to be set for devices that support TCP/IP communications. The TCP/IP communication settings include the IP address of the device and the port where the device listens for connections.

4.1.1.15 Set Telephony Communication Settings (Use Case)

The system allows Telephony communications settings to be set for devices that communicate via a Telephony Card, such as some HAR and SHAZAM device models. The Telephony settings include the default phone number, one or more port managers, the phone number to use when connecting to the device from each selected port manager, and the port manager connection timeout.

4.1.2 ControlHAR (Use Case Diagram)

The system allows users to control Highway Advisory Radio (HAR) devices deployed throughout the state to broadcast traffic alerts to motorists. The system also allows roadside signs, known as SHAZAMs to be activated to notify travelers to tune their radio to a specified station to hear the traffic alert that is being broadcast. In R8 support for TCP/IP communications is added for the DR1500 HAR and a new SHAZAM model is added. The underlying software changes will affect most (if not all) use cases shown on this diagram.

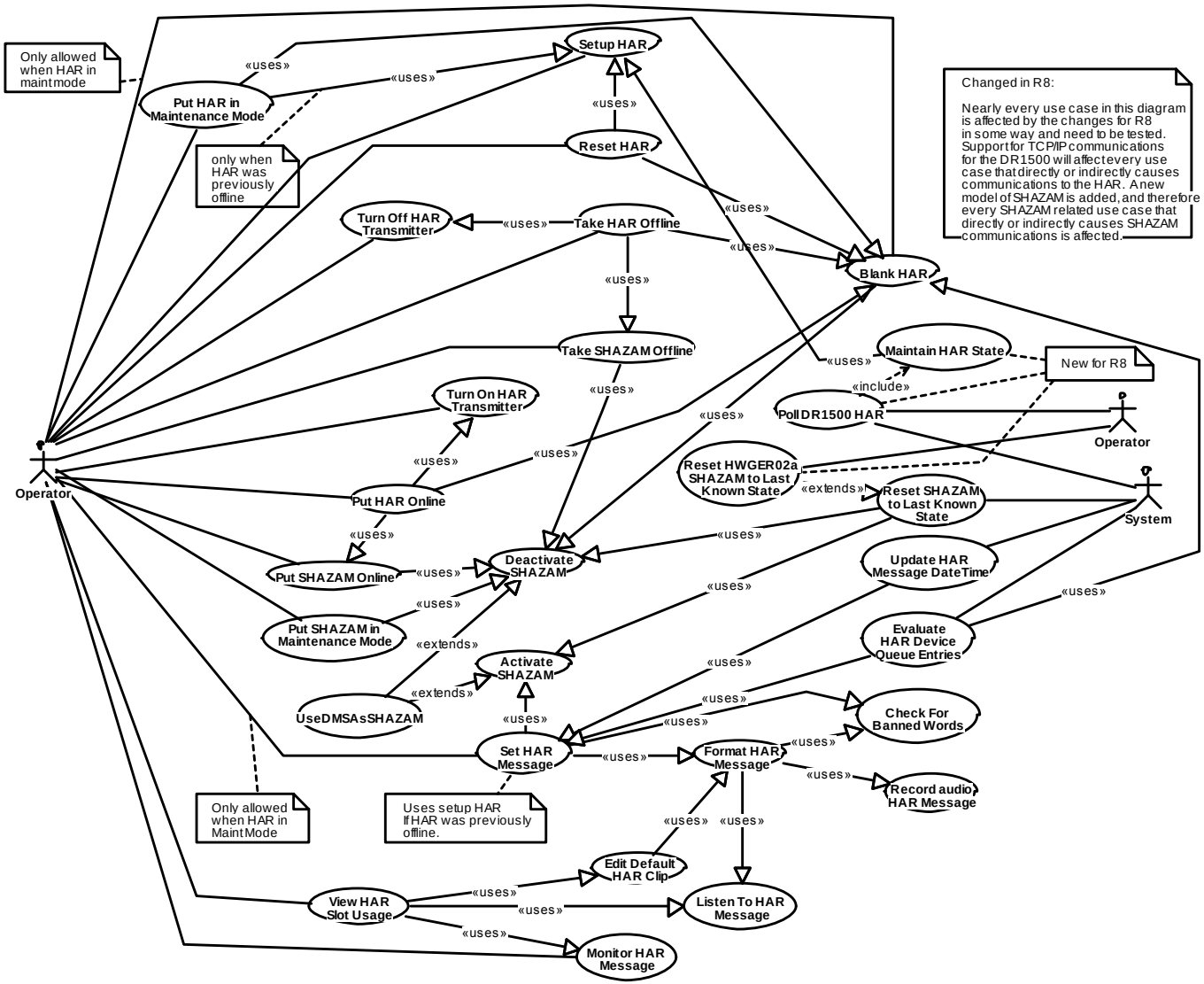


Figure 4-2. ControlHAR (Use Case Diagram)

4.1.2.1 Activate SHAZAM (Use Case)

An online SHAZAM is activated through a HAR message activation that includes the SHAZAM. A SHAZAM may also be activated independently when it is in maintenance mode. When a SHAZAM device is activated, its beacons are enabled. In the case of a DMS acting as a SHAZAM, a previously configured message (similar to a message that would be displayed on a SHAZAM with a fixed sign) is displayed.

An online SHAZAM can only be activated if the SHAZAM is associated to a HAR and the HAR is currently playing a message (other than the default message).

If a SHAZAM is allowed to be activated when it is already in use by an event response plan (same op center usage or override functional right), a message is logged in the original event's history indicating that the SHAZAM is no longer in use by the event.

4.1.2.2 Blank HAR (Use Case)

A HAR can be blanked if it is online or in maintenance mode. When the HAR is online, the device is only blanked if there are no traffic events that have currently requested that a message be placed on the device. When the HAR is in maintenance mode, the HAR can be blanked directly by the user.

A HAR can be blanked indirectly through administrative functions such as placing the device online or resetting the device.

When a HAR is blanked, the system will set the HAR's default message to be the current message. Additionally, the system will deactivate any associated active SHAZAMs before blanking the HAR itself.

This functionality is being updated in R8 to support TCP/IP communications to DR1500 devices.

4.1.2.3 Check For Banned Words (Use Case)

An operator (or the system) validates a text message by checking the words against the list of banned words for a particular device type. The check for banned words will be case insensitive.

4.1.2.4 Deactivate SHAZAM (Use Case)

An online SHAZAM can be deactivated by a user closing an event that was using the SHAZAM in its response plan or indirectly through operations such as taking a SHAZAM or HAR offline. A SHAZAM can also be deactivated independently

when the SHAZAM is in maintenance mode. Deactivating a SHAZAM stops its beacons from flashing and in the case of a DMS acting as a SHAZAM, blanks the "tune radio" message. When a SHAZAM is deactivated and it was being used in an event's response plan, a message is logged in the event's history indicating the SHAZAM was deactivated.

4.1.2.5 Edit Default HAR Clip (Use Case)

A user can edit the default header, trailer, and message stored in the HAR controller.

4.1.2.6 Evaluate HAR Device Queue Entries (Use Case)

The system shall evaluate entries placed on a HAR's arbitration queue in response to traffic events. The system shall use a priority algorithm to determine which message shall be placed on the HAR device. The system shall evaluate entries when a new entry is added, when an entry is removed, and when notified by the HAR device object that a previous asynchronous request has completed. When the queue is evaluated, the highest priority message shall be set on the HAR device, unless it is already currently set on the HAR device. When an evaluation occurs and the queue has become empty, the queue shall set the HAR device to its default message. The queue shall allow the concatenation of multiple messages to be set on the HAR device as the recording space on the HAR allows and according to configuration settings and concatenation rules. The rules that govern this feature are specified by an administrator in the system profile.

4.1.2.7 Format HAR Message (Use Case)

An operator may use the HAR message editor to create a HAR message. The editor will allow the operator to enter header, body and trailer text for the message. The text will be validated for banned and approved words. The editor will also allow the operator to view the run-time of the spoken message in minutes and seconds. If the run-time is greater than two minutes the system will alert the user by displaying the run-time in red text. The editor will allow an operator to insert delays between message segments. Text messages created by the user will be converted to an audio format by the system.

Message text shall allow inclusion of an optional date/time field that can be automatically updated by the system. This field can be included in-line in the text and may be used more than once in the message. The date/time field shall specify the format of this field (when included). Valid formats shall include general time of day (morning 00:00 - 11:59, afternoon 12:00 - 16:59, evening 17:00 - 23:59) and others (TBD). The system shall replace the date time fields with text based on the current time of day and the specified format.

4.1.2.8 Listen To HAR Message (Use Case)

The user can listen to the current message CHART has specified for broadcast by a HAR device. The user can listen to the header, body, and/or trailer. Note that the actual message being broadcast by the HAR could differ if the device is failed or has been commanded outside of the CHART system.

4.1.2.9 Maintain HAR State (Use Case)

DR1500 HARs using TCP/IP communications will be able to use poll data to correct the state of a HAR if it's detected to be out of synch with the state specified in CHART. Out of synch state will be detectd by querying the current play list and transmitter status.

4.1.2.10 Monitor HAR Message (Use Case)

The system allows the user to monitor the actual audio being broadcast by a HAR. The user may also monitor the actual audio stored in the controller for the default header, trailer, or message clips. This feature is only available if the HAR model and communications method supports monitoring, which includes the AP55 and the HIS DR1500 when a telephony port is used for communications. This feature is not available for the DR1500 when TCP/IP communications are used.

4.1.2.11 Operator (Actor)

An operator is a user of the system who has been assigned a valid username/password combination and granted roles for system access.

4.1.2.12 Poll DR1500 HAR (Use Case)

The ability to poll DR1500 HARs will exist for devices configured for TCP/IP communications. Polling will be done automatically on a specified period if automatic polling is enabled for the HAR. The user can force a poll if the HAR is online and the user possesses the maintain HAR or respond to event right, or if the HAR is in maintenance mode and the user possesses the maintain HAR right. Asside from collecting status data for display on the HAR's details page, the polling process will also ensure the state of the HAR is consistent with the state as last commanded by the system (see Maintain HAR State) and will detect if a hardware failure condition is present (see Set DR1500 Hardware Failure Detection Settings). The system shall consider a DR1500 hardware failed if any of the status values fall outside the configured hardware failure detection settings or if the HAR controller is found to have the power off. If a communication or hardware failure is detected while polling the HAR, alerts and/or notifications will be sent to the configured operations center and notification group as set

for the HAR (if any - see Set Alert and Notification Settings). Notifications will be sent to the configured group(s) (if any) on any operational status transition (OK to hardware failed, OK to communication failed, communication failed to OK, communication failed to hardware failed, hardware failed to OK, and hardware failed to communication failed).

4.1.2.13 Put HAR in Maintenance Mode (Use Case)

A user with proper functional rights may place a HAR in maintenance mode. When placed in maintenance mode, if the HAR was previously offline, the setup command is used to reload the HAR's slots that are configured for use in CHART. If the HAR was previously online, the HAR's message is set to its default message. The HAR shall proceed to maintenance mode even if attempts to control the device during this process fail. When a HAR is placed in maintenance mode, the controlling op center of the HAR becomes the op center of the user that performed the operation.

4.1.2.14 Put HAR Online (Use Case)

A user with appropriate privileges can put a HAR device online if it has previously been taken offline or put in maintenance mode. This automatically turns on the HAR transmitter and makes the HAR available for control through the system. When a HAR is placed online, the user shall be given the option to put any associated SHAZAMs online as well.

4.1.2.15 Put SHAZAM in Maintenance Mode (Use Case)

A user with proper functional rights can place a SHAZAM in maintenance mode if the SHAZAM is online or offline. When the SHAZAM is placed in maintenance mode, an attempt is made to deactivate the SHAZAM. Even if this attempt fails, the SHAZAM proceeds to maintenance mode and the controlling operations center of the SHAZAM is set to the operations center of the user that performed the operation.

4.1.2.16 Put SHAZAM Online (Use Case)

A user with appropriate privileges can put a SHAZAM online if the SHAZAM is currently offline or in maintenance mode. Putting the SHAZAM online makes it available for control through the system.

4.1.2.17 Record audio HAR Message (Use Case)

A user with appropriate privileges can record an audio message as an alternative to entering a text message. The operator's voice will be recorded in a binary audio file format using configurable system wide audio settings. The audio format and default settings have not yet been determined. These system wide voice recording audio setting values shall match those used

in the text to speech conversion. Manually recorded audio will require the user to enter a description of the message to be used in status displays.

4.1.2.18 Reset HAR (Use Case)

A user with proper privileges can reset a HAR that is in maintenance mode. Resetting a HAR clears the HAR's memory and restores it to its factory settings. All messages previously stored in the HAR controller are lost from the controller. The system automatically issues the setup command after the HAR is reset to restore the settings and to restore the messages that were previously stored in the controller.

4.1.2.19 Reset HWGER02a SHAZAM to Last Known State (Use Case)

During the refresh operation on a SWG-ER02a SHAZAM, the system shall first query the current state of the device to determine if its beacons are active or not active. If the beacons are currently in the state as desired by the system, no further action is required. If not, the system shall set the state of the beacons to the state desired by the system and re-query the device to determine if the device is in the proper state. If not, the system shall indicate the SHAZAM is hardware failed.

4.1.2.20 Reset SHAZAM to Last Known State (Use Case)

The system will periodically connect to each SHAZAM and issue a command to put the SHAZAM in the state as indicated by the system as the last known state. This is a safe guard put in place for SHAZAM models that cannot be polled to ensure their state remains consistent with the state desired by the system. If communications to the SHAZAM are detected to be failed the system may issue an alert and/or a notification as specified in the alert and notification settings for the SHAZAM. If the SHAZAM model supports detecting hardware failures, this detection shall be performed as part of resetting the SHAZAM to the last known state. If a communication or hardware failure is detected while restting the SHAZAM to the last known state, alerts and/or notifications will be sent to the configured operations center and notification group as set for the SHAZAM (if any - see Set Alert and Notification Settings). Notifications will be sent to the configured group(s) (if any) on any operational status transition (OK to hardware failed, OK to communication failed, communication failed to OK, communication failed to hardware failed, hardware failed to OK, and hardware failed to communication failed).

4.1.2.21 Set HAR Message (Use Case)

A HAR's message is set through the execution of an event response plan or set directly by an administrator when the device is in maintenance mode. The message activation may specify messages which were previously stored in message slots in the

controller or a message that was created using the HAR message editor.

When activating a HAR message created by the message editor the user may choose to use the default header and trailer or just use the message body for the entire message. Messages activated in this manner shall be loaded into the HAR controller in the slot designated for immediate broadcast.

A HAR message activation also specifies if each associated SHAZAM should be activated or not. The selected notifiers will be activated only after the message has been activated on the HAR.

The system shall support sending messages to at least 4 HARs at one time; each constituent of a synchronized HAR counts as 1 HAR toward this total. A synchronized HAR is comprised of individual constituent HARs that play the same message at the same time. Each constituent can be specified as being active or inactive; messages activated on a synchronized HAR are only activated on the active constituents.

4.1.2.22 Setup HAR (Use Case)

An administrator can issue the setup command on a HAR that is in maintenance mode. The setup command causes the CHART II system to load its configuration values for the HAR into the device. The setup command also causes all messages that are currently specified to be stored in the HAR controller to be restored into the device.

4.1.2.23 System (Actor)

The System actor represents any software component of the CHART system. It is used to model uses of the system which are either initiated by the system on an interval basis, or are an indirect by-product of another use case that another actor has initiated.

4.1.2.24 Take HAR Offline (Use Case)

A user with appropriate privileges can take a HAR offline to disallow control of the HAR through the system. When a HAR is taken offline, the HAR's transmitter is turned off and all associated SHAZAM devices are also taken offline.

4.1.2.25 Take SHAZAM Offline (Use Case)

A user with appropriate privileges can take a SHAZAM offline if it is online or in maintenance mode. A SHAZAM that has been taken offline is not able to be controlled through the system (activated or deactivated) until the SHAZAM is put online.

Taking a SHAZAM off line does not affect any HAR that has been associated with the SHAZAM.

4.1.2.26 Turn Off HAR Transmitter (Use Case)

A user with appropriate privileges can turn off the transmitter of a HAR that is in maintenance mode.

4.1.2.27 Turn On HAR Transmitter (Use Case)

A user with appropriate privileges can turn on the transmitter of a HAR that is in maintenance mode.

4.1.2.28 Update HAR Message DateTime (Use Case)

The system shall periodically update HAR messages that are currently active and contain a date/time field. The date/time field shall be updated based on the current time of day and the format specified in the date/time field.

4.1.2.29 UseDMSAsSHAZAM (Use Case)

A user with appropriate privileges may opt to control DMS as a SHAZAM for a HAR. In such a case, a pre-configured message is set on the DMS that informs the traveler about the HAR message being broadcast, with the DMS beacons flashing.

4.1.2.30 View HAR Slot Usage (Use Case)

A user may view the current HAR controller slot usage for the default header, default trailer, and default message. This shall include the message that is stored in the slot, and the total time used by all messages stored in the controller, the total time supported by the controller, and the total recording time remaining. The system shall allow the user to listen to the default header, default, trailer, and default message. The system shall allow the user to monitor (listen to the actual audio as played from the HAR controller) the default header, default trailer, and default message if the HAR is an AP55 HAR or the HAR is a DR1500 HAR that uses Telephony communications.

4.1.3 ViewHARandSHAZAM (Use Case Diagram)

This diagram shows use cases related to viewing HAR and SHAZAM devices within the system.

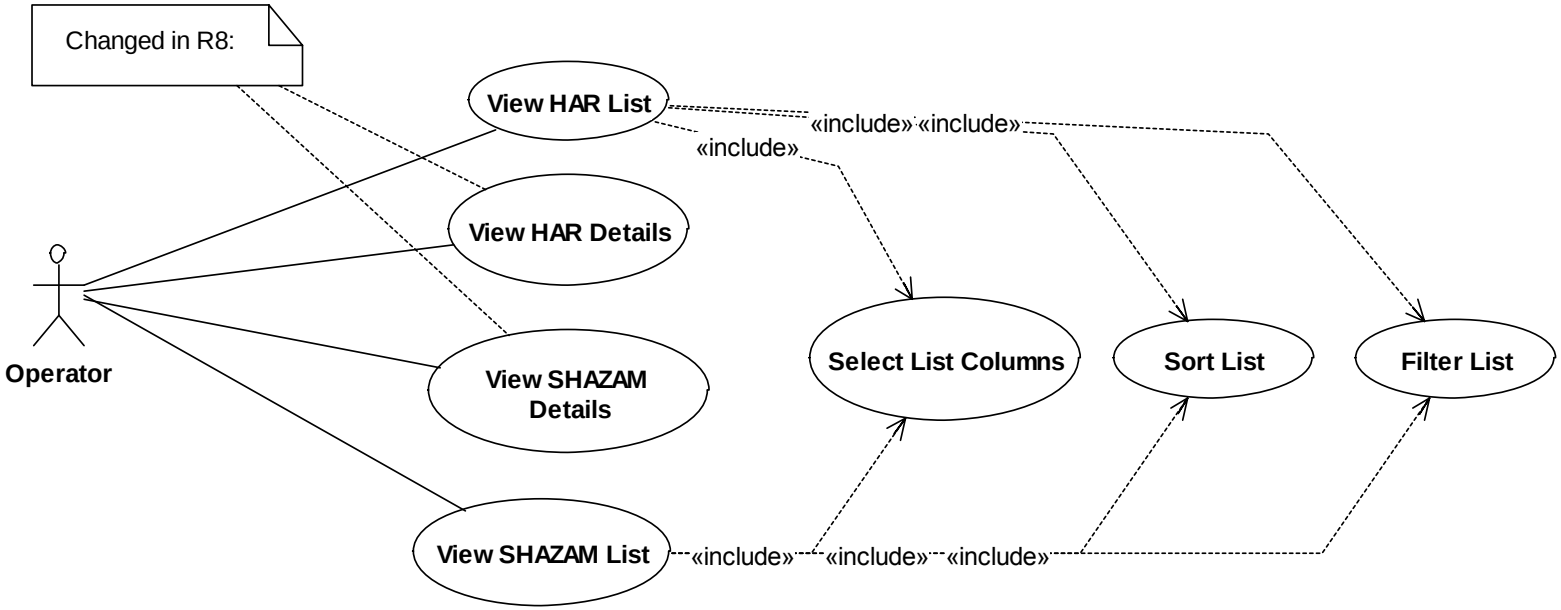


Figure 4-3. ViewHARandSHAZAM (Use Case Diagram)

4.1.3.1 Filter List (Use Case)

The system allows lists to be filtered to include only rows that have specified values in a column. Multiple filters can be used on a single list, and the system allows all filters to be removed to view all items in the list. Columns that have a large range of possible values may not support filtering (for example a column that displays a device message).

4.1.3.2 Operator (Actor)

An operator is a user of the system who has been assigned a valid username/password combination and granted roles for system access.

4.1.3.3 Select List Columns (Use Case)

The system allows the user to choose which columns are to appear in a list and which columns are to be hidden. Certain columns such as the name of an item may not be permitted to be hidden.

4.1.3.4 Sort List (Use Case)

The system allows lists to be sorted using values in a specific column of the list. Ascending and descending sorts are supported. Not all columns support sorting.

4.1.3.5 View HAR Details (Use Case)

The system shall allow a user with appropriate rights to view the details of a HAR device. The details will include the current message, traffic events using the HAR (if any), associated notifiers (if any), and the status. The status includes the HAR transmitter state (on/off), the mode (online, offline, maint mode) and operational status (OK, Comm. Failed, Hardware Failed). The details shall also include the current message clips stored as the default header, default trailer, and default message. The recording capacity status shall be shown and also a timestamp that shows the last time a setup command has been successfully executed on the HAR. The details page also contains configuration information for the HAR, which will include the currently selected communication method (Telephony Port or TCP/IP). When telephony port communications are selected, the details will include the default phone number for the device, the access code, port manager timeout, and the list of port managers and associated phone numbers used to access the HAR via each selected port manager. When TCP/IP communications are selected, the IP address and port shall be shown. Device phone numbers, access codes, port manager timeouts, port type, and IP addresses/ports are considered sensitive data and shall only be shown if the user posesses the proper user right to view the sensitive data for the device (based on the device's owning organization). Additional status and configuration information is included for DR1500 HARs that use TCP/IP communications. The additional status information includes the timestamp when the HAR was last found to have a status mismatch with its last commanded state. Other details include the power status, the DC voltage, broadcast monitor percent, mode, sub-mode, sync mode, DCC module version information, HAR module version information, HAR timestamp, and transmitter status. The transmitter status of a DR1500 includes set power, forward power, reflected power, voltage standing wave ratio (VSWR), and modulation percent. The additional configuration information shown for a DR1500 HAR that uses TCP/IP communications includes an indicator that shows if automatic polling is enabled, the polling interval, the alerts and notification settings, and the hardware detection failure settings. The hardware failure detection settings include the minimum voltage and maximum VSWR.

4.1.3.6 View HAR List (Use Case)

The system will allow a user with appropriate rights to view the list of HARs defined in the system. Data shown for each HAR will include the name, location, current message, status, list of active notifiers (if any), list of traffic events using the HAR (if any), route, county, direction, milepost, owning organization, maintaining organization, port managers, frequency, call sign, connection site, and an indicator of whether or not the HAR is to be displayed on the map. To save screen space, the visible columns will be selectable. Several columns will be hidden by default to save space. The user will be able to sort the HAR list by any of the columns listed above. The user will be able to filter the HAR list by any of the columns listed above except name, location, and milepost.

4.1.3.7 View SHAZAM Details (Use Case)

The system shall allow a user with appropriate rights to view the details of a SHAZAM device. The details shown shall include the name, model (Viking RC2A or HWG-ER02a), static message displayed on the SHAZAM sign, owning organization, maintaining organization, network connection site, location, and communication settings. The communication settings shall indicate if a telphony port or tcp/ip is used to communicate with the SHAZAM. When a telephony port is used, the system shall display the default phone number, access code, port manager timeout and port managers (with phone numbers to be used to connect to the SHAZAM from the port manager). When TCP/IP is used, the system shall display the IP address and port used by the SHAZAM. Device phone numbers, access codes, port manager timeouts, port type, IP addresses and ports are considered sensitive information and will only be shown if the user has the right to view sensitive information for the SHAZAM (based on its owning organization). The auto-refresh settings shall be shown for the SHAZAM, indicating if auto-refresh is enabled and the interval at which auto-refresh is performed. The details will also include the relay being used by a SWG-ER02a SHZAM.

4.1.3.8 View SHAZAM List (Use Case)

The system will allow a user with appropriate rights to view the list of SHAZAMs defined in the system. Columns will include the name, location, associated HAR, beacon state, operational status, last update time, route, county, direction, milepost, owning organization, maintaining organization, port managers, connection site, and the show on map indicator. To save screen space, the visible columns will be selectable. Several of the columns will be hidden by default to save space. The user will be able to sort the SHAZAM list by any of the displayed columns. The user will be able to filter the SHAZAM list by any of the columns except name, location, associated HAR, last update time, and state milepost. The port managers column will only contain data for SHAZAMs set to use Telephony port communications.

5 Detailed Design – IP HAR and IP SHAZAM

5.1 Human-Machine Interface

5.1.1 HAR

The human-machine interface for HAR devices is changed in R8 to allow TCP/IP communications to be specified for DR1500 HAR devices. This includes the ability to enable polling for DR1500 HARs that use TCP/IP communications, and the ability to set thresholds that determine when the system should consider a DR1500 HAR hardware failed when polled. Additionally, device failure alerts and notifications can be enabled for HAR devices. See the sections below for details on portions of the HAR human-machine interface that are new or changed for R8.

5.1.1.1 Add HIS DR1500 HAR

When viewing the HAR list, a new HIS DR1500 HAR can be added to the system by clicking the Add HIS DR1500 HAR link near the top of the page (see below).

Figure 5-1 Add HIS DR1500 HAR Link

Several sections of the form used to add a HIS DR1500 HAR to the system have been changed for R8, and two sections have been added to the form. See below for a description of each section the Add HIS DR1500 HAR form.

5.1.1.1.1 General HAR Information

There are no changes to this section of the ADD HAR form for R8.

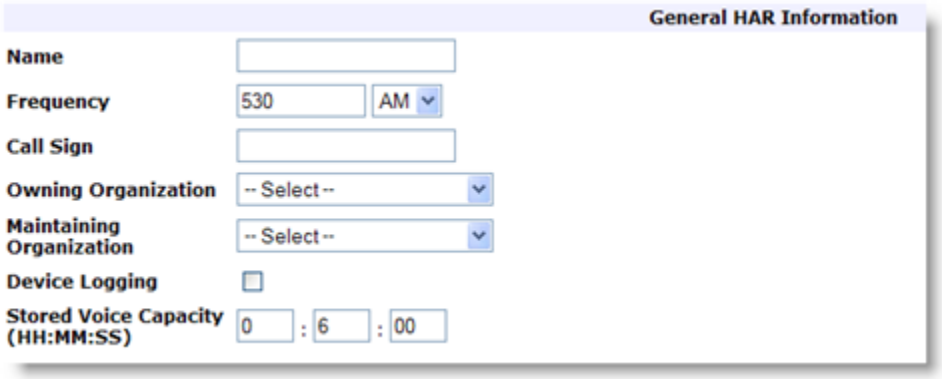


Figure 5-2 Add HAR - General HAR Information

5.1.1.1.2 Location

There are no changes to this section of the Add HAR form for R8.

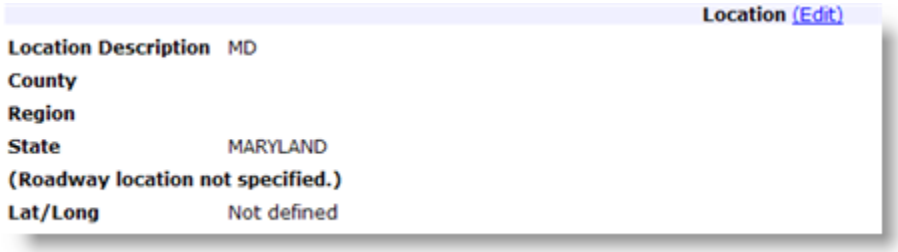


Figure 5-3 Add HAR - Location

5.1.1.1.3 Default Clips

There are no changes to this section of the Add HAR form for R8.

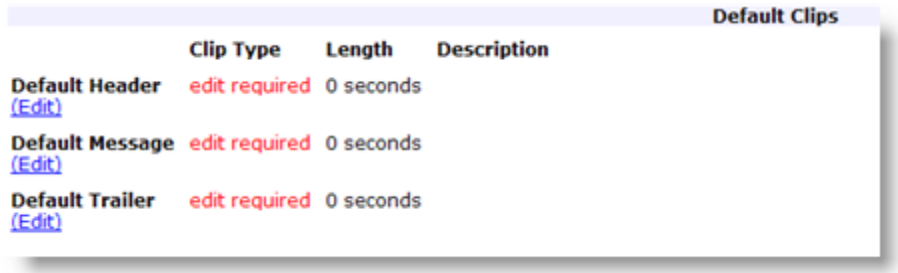


Figure 5-4 Add HAR - Default Clips

5.1.1.1.4 Device Control Communications

This section of the Add HAR form is changed for R8 to allow Telephony or TCP/IP communications to be selected for a DR1500 HAR. When telephony port communications are

selected, this section of the form contains fields that apply to telephony communications. There are no changes to the telephony port fields for R8 (see below).

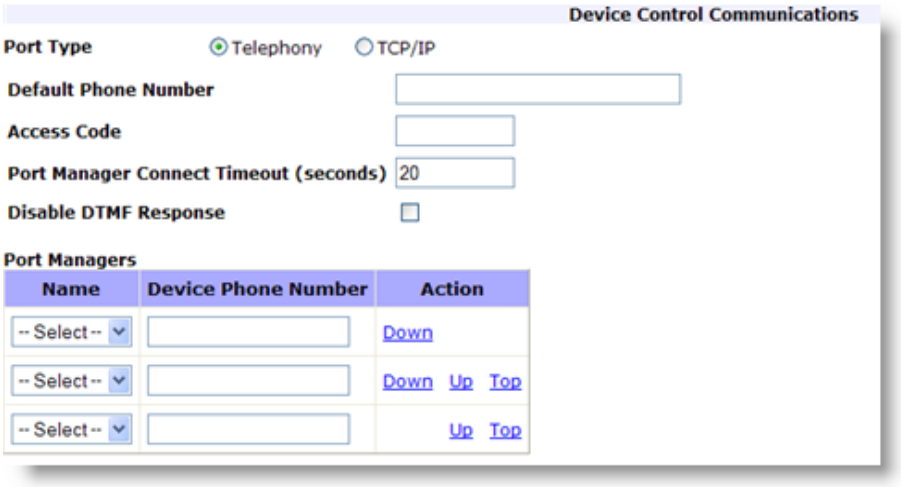


Figure 5-5 Add HAR - Device Control Communications, Telephony

When a port type of TCP/IP is selected, this section of the form changes to include fields that apply to TCP/IP communications. This is new for R8 (see below).

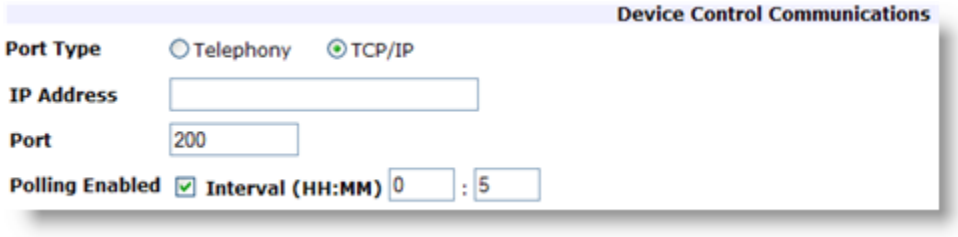


Figure 5-6 Add HAR - Device Control Communications, TCP/IP

The IP Address is the address as configured for the HAR device. The Port is the port where the HAR device listens for connections. The default port for a DR1500 HAR is 200. When TCP/IP communications are used for a DR1500, polling can be enabled, when when polling is enabled the polling interval can be specified. The default polling interval is 5 minutes. When device polling is enabled, CHART will connect to the HAR periodically and query status values from it. CHART will inspect the status values to attempt to determine if the HAR is broadcasting the message as specified in CHART (or is blank as specified in CHART). If CHART determines the device is not broadcasting the desired message (or is not blank as desired), CHART will automatically issue a setup command to the HAR to set its message and transmitter status as last commanded by CHART. During polling, CHART will also attempt to detect if the HAR is hardware failed, and if so will change the operational status of the HAR to hardware failed in CHART.

5.1.1.1.5 Associated Message Notifiers

There are no changes to this section of the Add HAR form for R8 (see below).

Figure 5-7 Add HAR - Associated Message Notifiers

5.1.1.1.6 DR1500 Hardware Failure Detection Settings

The DR1500 Hardware Failure Detection Settings section is new for R8. This section only appears when the port type in the control line communication settings section is set to TCP/IP because status can only be obtained from the DR1500 when using TCP/IP communications. These settings are thresholds used by CHART to determine if the status values obtained while polling the HAR are within acceptable ranges, and if not, CHART will set the HAR to an operational status of hardware failed.

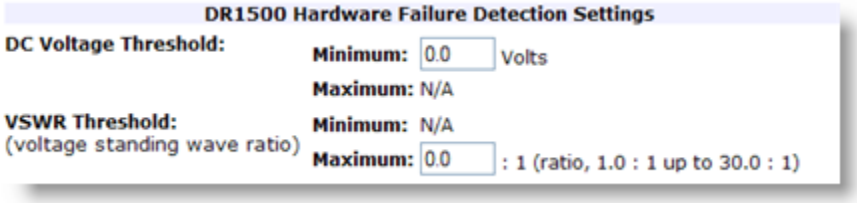


Figure 5-8 Add HAR - DR1500 Hardware Failure Detection Settings

The DC Voltage Threshold is the minimum DC voltage allowed. If a status poll detects a value lower than this, the HAR will be considered hardware failed. The voltage standing wave ratio threshold is the maximum allowable ratio. If the HAR device reports a ratio higher than the value specified, CHART will consider the HAR hardware failed.

5.1.1.1.7 Alert and Notification Settings

The alert and notification settings section of the Add HAR form is new for R8. This section allows device failure alerts and notifications to be enabled for the HAR. Two types of device failure are supported in the CHART system, communications failure and hardware failure. The alert and notification settings section will appear for all models of HAR, despite their method of communication (Telephony or TCP/IP). Only the DR1500 HAR when operated with TCP/IP communications will be capable of becoming hardware failed, and therefore the hardware failure alert and notification selections will not have an effect on other HARs present at the time R8 is released. There is no harm setting these values for other HARs, however, and if at some point in the future CHART becomes capable of detecting hardware failures for those HARs, the hardware failure settings in this section of the form would in fact be used.

Figure 5-9 Add HAR - Alert and Notification Settings

The Operations Center to alert on Comm Failure is just as the field states; when a communications failure is detected for the HAR, an alert will be raised and assigned to the specified operations center if any. When set to "None", comm failure alerts for the HAR are disabled. The Operations Center to alert on Hardware Failure is also just like its name states; when a harware failure is detected for the HAR, an alert will be raised and assigned to the selected operations center, if any. When set to "None", hardware failure alerts for the HAR are disabled. The Notification group to notify on Comm Failure is the group that will receive a notification when the HAR becomes comm failed. The selected group will also receive notification when the HAR transitions from Comm Failed to OK, or from Comm Failed to Hardware Failed. When set to "None", all comm failure notifications for the HAR are disabled. The Notification group to notify on Hardware Failure is the group that will be notified if the HAR becomes hardware failed. The selected group is also notified if the HAR transitions from Hardware Failed to OK, or from Hardware Failed to Comm Failed. Hardware failure notifications are disabled if the selection is set to "None".

5.1.1.1.8 Site Selection

There are no changes to this section of the Add HAR form for R8.

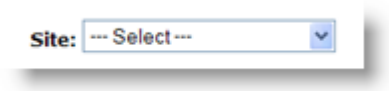


Figure 5-10 Add HAR - Site Selection

5.1.1.2 HAR Details

5.1.1.2.1 Actions

The Actions section of the HAR details page is changed for R8 to include a "Poll Now" link for the HAR if the HAR is a DR1500 that is configured to use TCP/IP communications and the HAR is online. Clicking the Poll Now link will cause the command status page to be shown where the results of the polling of the device will be shown.

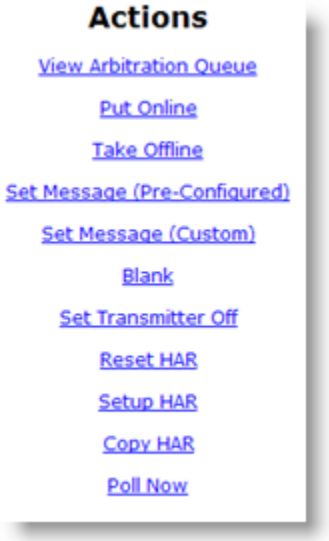


Figure 5-11 HAR Details - Actions

5.1.1.2.2 Message

The Message section of the HAR details page contains no changes for R8.

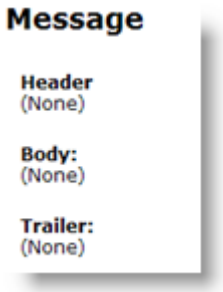


Figure 5-12 HAR Details - Message

5.1.1.2.3 Used By

The Used By section of the HAR details page is unchanged in R8.

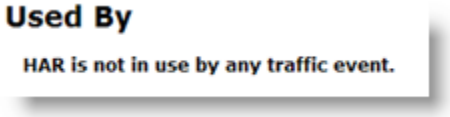


Figure 5-13 HAR Details - Used By

5.1.1.2.4 Associated Message Notifiers

There are not changes to the Associated Message Notifiers section of the HAR Details page for R8.

Figure 5-14 HAR Details - Associated Message Notifiers

5.1.1.2.5 Status

The status section of the HAR details page is updated in R8 to add new status fields. The following existing fields are unchanged for R8: Controlling Center, Transmitter Status (on/off), Mode, Status, Last Status Change, and Last Contact. The Last Setup field is new for R8 and applies to all HARs. This is the last time a setup command was successfully issued to the HAR. The remaining fields are all new for R8 and only apply to the DR1500 HAR, and only when the DR1500 is operated using TCP/IP communications. The screen shot below is for a DR1500 HAR that is set to use TCP/IP communications.

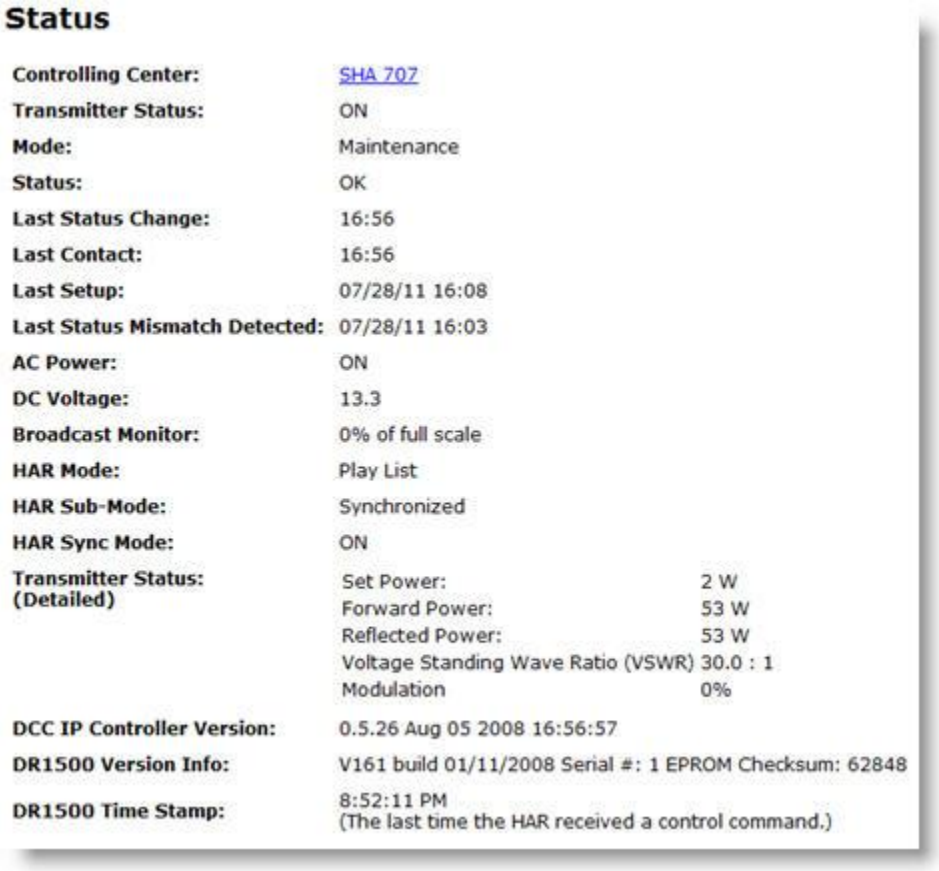


Figure 5-15 HAR Details - Status

The Last Status Mismatch Detected field is the time stamp that indicates the last time the HAR was polled and CHART determined its message / transmitter on/off status did not match the status as CHART last commanded. When a status mismatch is detected, CHART automatically issues a setup command for the HAR. The Power field lists the status of the power of the DR1500 and can be either ON or OFF. The DC Voltage is the voltage reading from the HAR. If this value as read from the HAR is outside the threshold as specified for the HAR in CHART, an indicator will be shown to indicate CHART considers the HAR hardware failed because of the value, and lists the threshold that was violated. The Broadcast Monitor field is the current broadcast monitor percent of full scale. The HAR Mode field indicates the mode in which the HAR is operating, which includes Off, Play List, Alert, Live, and Aux. This value will normally be set to Play List. The HAR Sub-Mode field indicates if the HAR is operating in a synchronized or unsynchonized mode. The HAR Sync Mode field indicates if conditions exist to allow the HAR to operate in synchronized mode. This value will be on or off. The transmitter status (detailed) fields show the transmitter related readings from the HAR; set power, forward power, reflected power, VSWR, and modulation percent. If the VSWR is outside the allowable theshold as configured for the HAR, a message will appear next to the value to indicate the reading is causing CHART to consider the HAR hardware failed. The DCC IP Controller Version field shows the firmware version information for the HAR's digital communications

controller, the component of the HAR that enables TCP/IP communications. The DR1500 Version Info contains version information for the main DR1500 component. The DR1500 Time Stamp is a time stamp read from the HAR that indicates the last time a control command was received by the HAR.

5.1.1.2.6 Clips Stored In HAR

There Clips Stored In HAR section of the HAR details page is changed for R8 such that the Monitor links (shown in the image below) will not appear if the HAR is a DR1500 that is configured to communicate using TCP/IP.

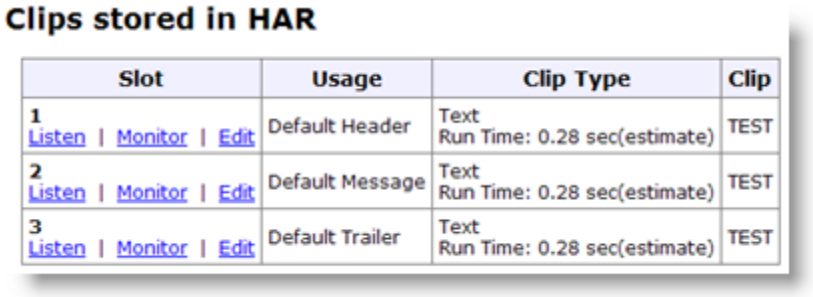


Figure 5-16 HAR Details - Clips Stored in HAR

5.1.1.2.7 Recording Capacity Status

There are no changes to the Recording Capacity Status section of the HAR details page for R8.

Figure 5-17 HAR Details - Recording Capacity Status

5.1.1.2.8 Configuration

There are no changes to the Basic Settings and Location sections of the HAR configuration portion of the details page.

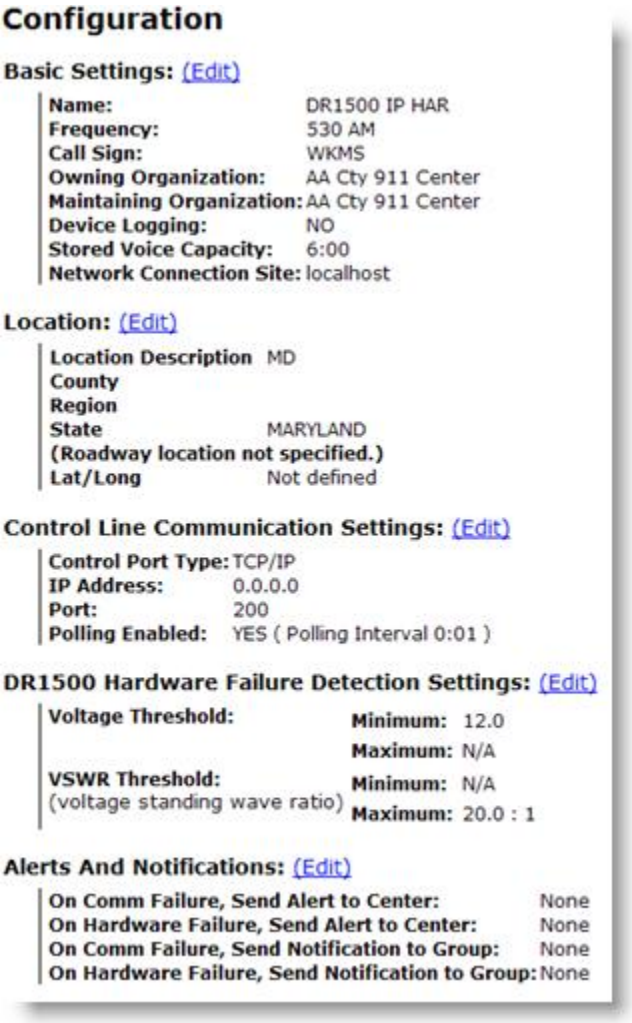


Figure 5-18 HAR Details - Configuration

The Control Line Communication Settings section is changed for R8 to show the port type (TCP/IP or Telephony) and to show the communication settings specific to the port type. When the port type is TCP/IP, polling settings are also shown. The DR1500 Hardware Failure Detection Settings section is new for R8 and applies only to the DR1500 HAR model, and only when configured to utilize TCP/IP communications. It shows the configured minimum and maximum values, as applicable, for the status reading from the HAR that are used to determine if CHART should consider the HAR hardware failed. The Alerts and Notifications section is also new for R8. This section applies to all HAR models regardless of communications port type. It shows the operations centers to be alerted of communication and hardware failures, and the notification groups to be notified of communication and hardware failures (if any).

5.1.1.3 Edit Control Line Communication Settings

The Edit Control Line Settings form is accessed by clicking the Edit link on the HAR details page next to the Control Line Communication Settings section heading:

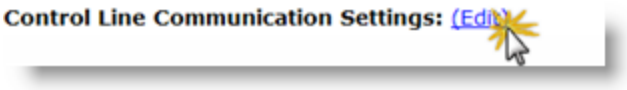


Figure 5-19 HAR Details - Edit Control Line Communication Settings Link

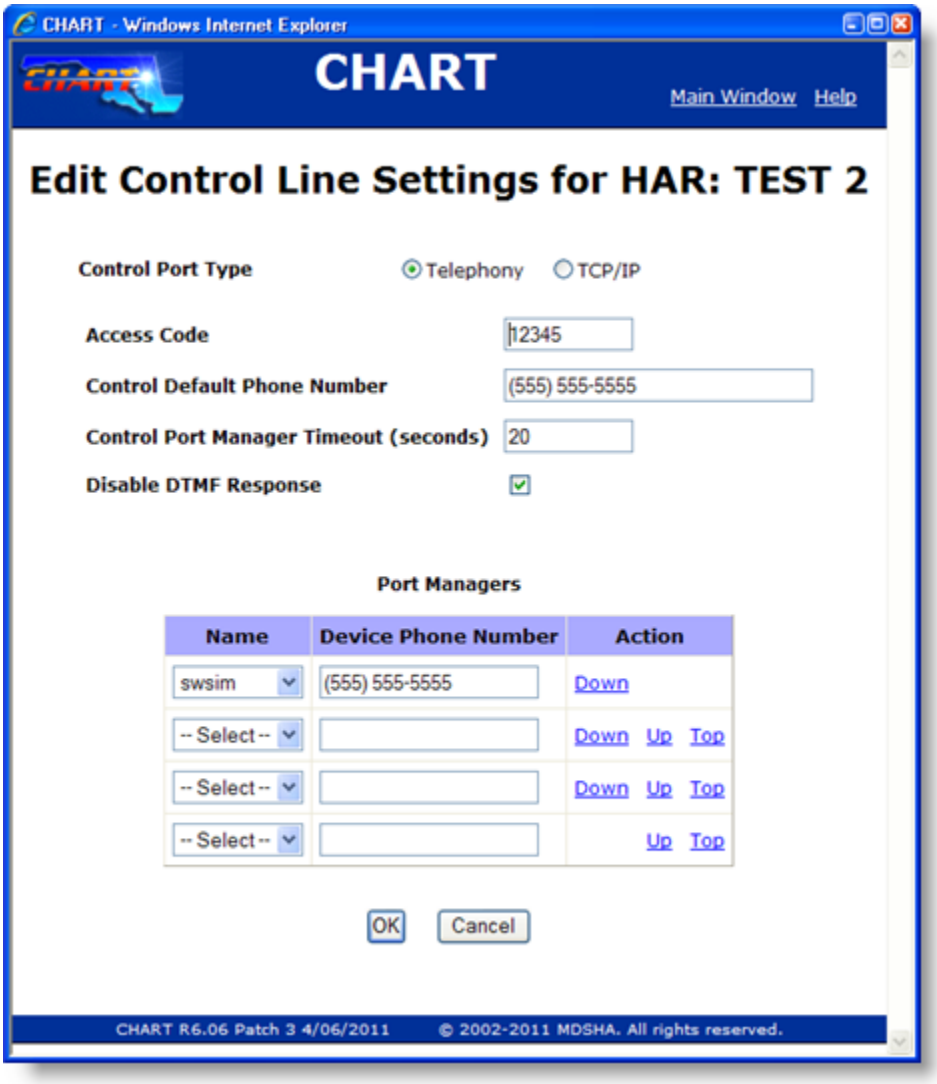


Figure 5-20 Edit HAR Control Line Settings Form, Telephony

This form is changed in R8 to allow selection of the Control Port Type. When Telephony is selected, as shown above, the form is the same as it was prior to R8. When TCP/IP is selected, the form changes as shown below:

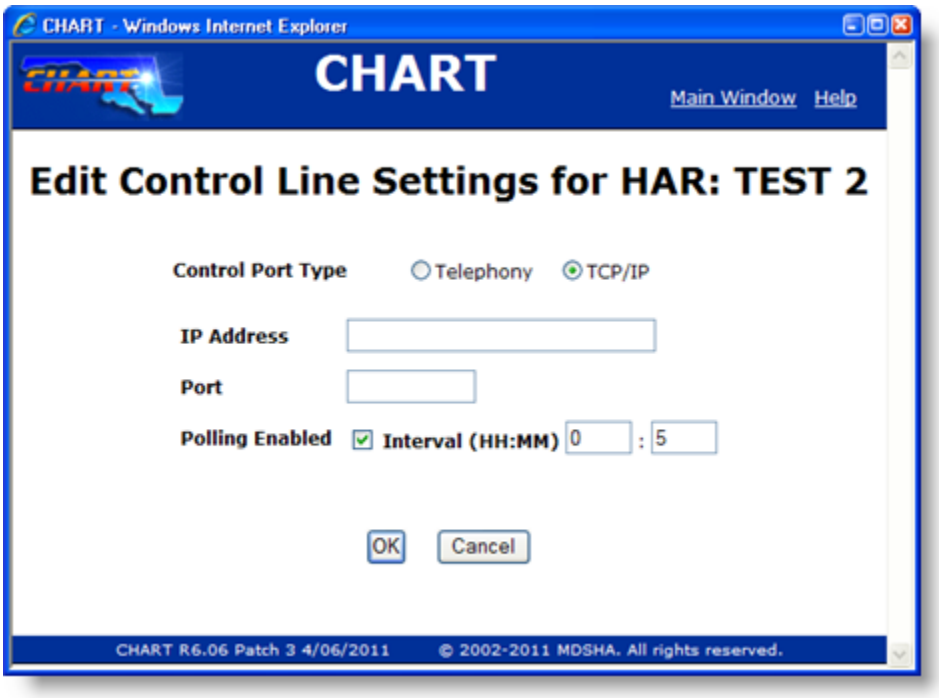


Figure 5-21 Edit HAR Control Line Settings Form, TCP/IP

When TCP/IP communications are selected (supported only for the DR1500 model), the form requires the IP Address and Port to be entered, and allows polling to be enabled. When polling is enabled, the polling interval must be specified, in minutes.

5.1.1.4 Edit DR1500 Hardware Failure Detection Settings

The DR1500 Hardware Failure Detection Settings form is accessed by clicking on the Edit link next to the DR1500 Hardware Failure Detection Settings section header on the HAR details page for a DR1500 HAR. Note that this section only appears if the DR1500 is set for control via TCP/IP.

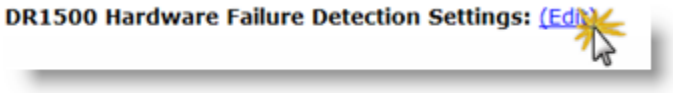


Figure 5-22 Edit DR1500 Hardware Failure Detection Settings Link

After clicking the Edit link, the following form is shown:

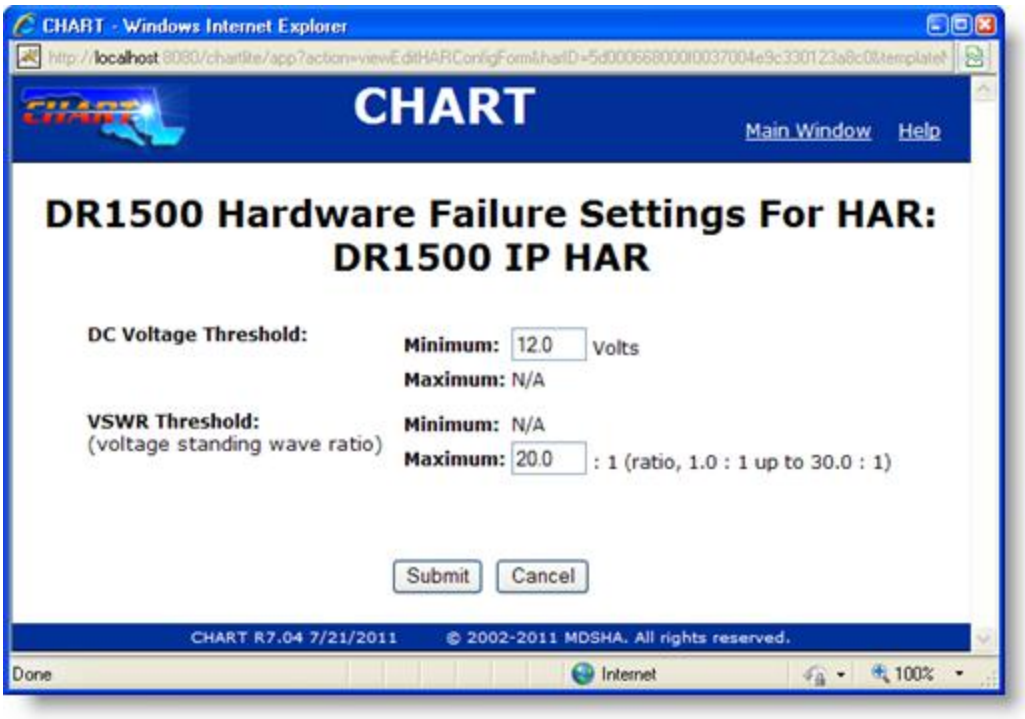


Figure 5-23 Edit DR1500 Hardware Failure Detection Settings Form

Each of the 4 readings from the HAR that are used by CHART to detect a hardware failure condition exists on this form. Minimum and Maximum values can be set as applicable to each type of reading. Note that VSWR is specified as the left hand side of a ratio, where the right hand side is always 1. So for example you could specify a ratio of 1:1, 2:1, 20:1, etc. by entering 1, 2, or 20 in the entry field (the :1 part of the ratio is fixed and can't be changed).

5.1.1.5 Edit Alert and Notification Settings

The alert and notification settings form is accessed by clicking the Edit link next to the Alerts and Notification section header on the HAR details page:

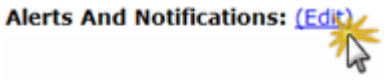


Figure 5-24 Edit HAR Alerts and Notifications Link

After clicking the Edit link, the following form is shown:

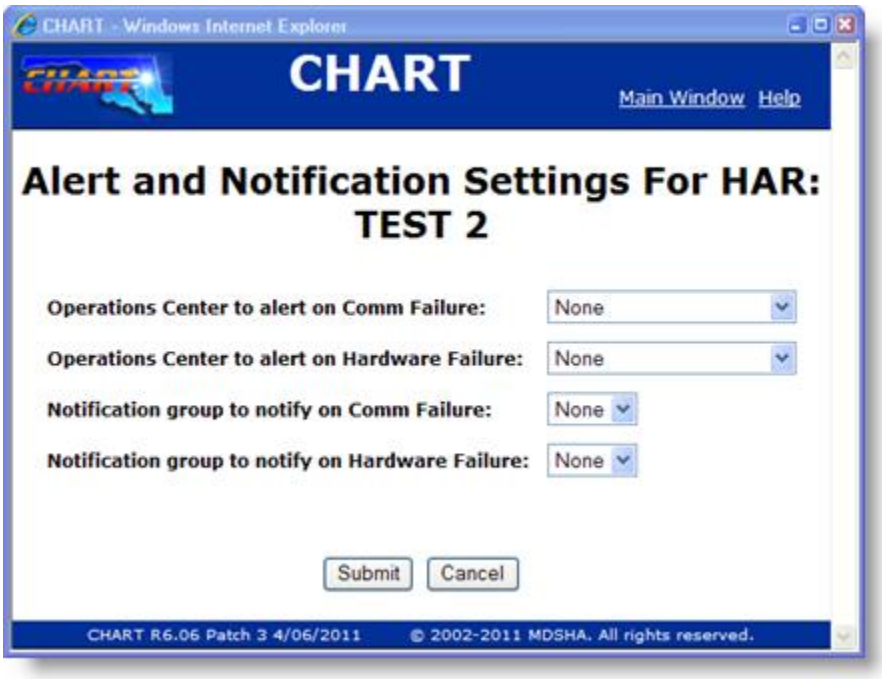


Figure 5-25 Edit HAR Alert and Notification Settings Form

This form allows selection of an operations center to receive comm failure alerts for the HAR and selection of an operations center to receive hardware failure alerts for the HAR. A selection of None is available to indicate that alerts of the given type should not be generated for the HAR. Similarly the form allows selection of the notification groups that are to receive comm failure and hardware failure notifications. Again, a selection of None can be used to disable that particular type of notifications. Note that notifications for HARs will be sent on each status transition (OK, Comm Failed, and Hardware Failed).

5.1.2 SHAZAM

5.1.2.1 Add SHAZAM

The Add SHAZAM form is accessed by click the Add SHAZAM link on the SHAZAM list page, as shown below:

Figure 5-26 Add SHAZAM Link

After clicking the Add SHAZAM link, the Add SHAZAM form is shown. Each section of this form is discussed in the sections below.

5.1.2.1.1 General SHAZAM Information

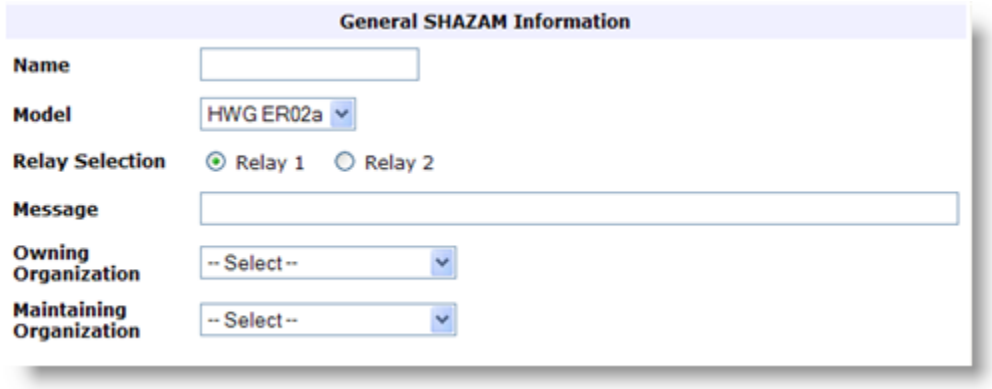


Figure 5-27 Add SHAZAM - General SHAZAM Information

The General SHAZAM information section of the Add SHAZAM form is changed in R8 to include a model select list. The available selections are Viking RC2A and HWG ER02a. The Viking RC2A was the only SHAZAM model supported by CHART prior to R8 and thus model selection was not needed. New for R8 is support for the HWG ER02a SHAZAM. When the ER02a SHAZAM model is selected the Relay Selection radio buttons (new for R8) appear to allow the user to specify which of the 2 electronic relays on the ER02a are connected to the beacons on the SHAZAM sign. The Relay Selection radio buttons do not appear on this form when the Viking RC2A model is selected.

5.1.2.1.2 Location

There are no changes to the Location section of the Add SHAZAM form for R8.

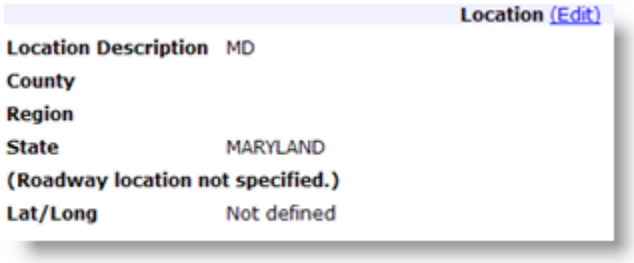


Figure 5-28 Add SHAZAM - Location

5.1.2.1.3 Device Communications

The Device Communications section of the Add SHAZAM form is changed for R8 to add the Port Type of TCP/IP. The Port Type field is not changeable because the Viking RC2A SHAZAM supports only Telephony port communications and the HWG-ER02a SHAZAM supports only TCP/IP communications. When the user selects the model, the Device Communications section will automatically change to communications port type supported by the SHAZAM model.

Figure 5-29 Add SHAZAM - Device Communications, Telephony

When the Viking RC2A model is selected, the Device Communications section of the Add HAR form (shown above) is unchanged for R8, except for the Port Type which is changed to a radio button in R8. (Prior to R8 it was just static text that said "Telephony".) When the HWG-ER02a model is selected, the Device Communications section automatically changes such that TCP/IP is selected, as shown below.

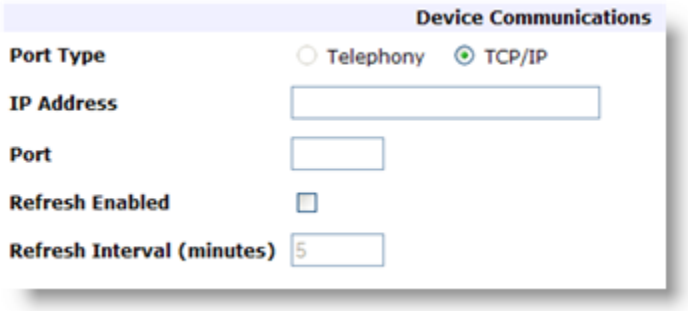


Figure 5-30 Add SHAZAM - Device Communications, TCP/IP

The Port Type of TCP/IP is new for R8. When TCP/IP is selected, the IP Address and Port must be specified. Refresh, which existed prior to R8 for Telephony port communications also applies to TCP/IP communications.

5.1.2.1.4 Alert and Notification Settings

The Alert and Notification settings section of the Add SHAZAM page is new for R8. It allows the operations centers to be set that are to receive an alert when CHART detects that the SHAZAM is comm failed or hardware failed. This section of the form also allows notification groups to be selected for both comm failure notifications and hardware failure notifications. A value of "None" is allowed to disable the alerts and/or notifications. Notifications related to SHAZAMs are sent on any status change of the SHAZAM; OK to Comm Failed, Comm Failed to OK, Comm Failed to Hardware Failed, OK to Hardware Failed, Hardware Failed to OK, and Hardware Failed to Comm Failed. Note that only the HWG-ER02a SHAZAM is capable of being hardware failed in R8 – the Viking RC2A does not support querying its status. If a Viking RC2A SHAZAM has hardware failure alerts or notifications enabled, this will not cause any problems in the system but these alerts and notifications will never be generated for that model SHAZAM.

5.1.2.1.5 Site

The Site section of the Add SHAZAM page is unchanged for R8.

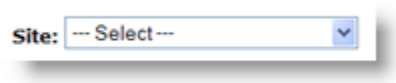


Figure 5-31 Add SHAZAM - Site Selection

5.1.2.2 SHAZAM Details

5.1.2.2.1 Actions

The actions section of the SHAZAM details page is unchanged for R8. When online the available actions (shown below) include the ability to refresh the SHAZAM by setting it to its last commanded state (activated or deactivated). When this is done for a HWG-ER02a SHAZAM (which supports querying its status), CHART has the the ability to detect if the SHAZAM is hardware failed, so it is possible that the SHAZAM's operational status may change after issuing this command.

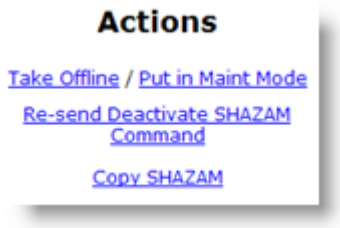


Figure 5-32 SHAZAM Details - Actions, Online

When the SHAZAM is in Maintenance Mode, the Actions section includes the ability to activate or deactivate the SHAZAM's beacons. Like the refresh command, when the activate or deactivate commands are issued CHART queries the SHAZAM after issuing the command to determine if it is indeed in the commanded state, and if not CHART will set the device to a status of hardware failed.

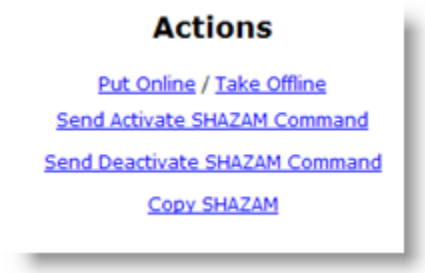
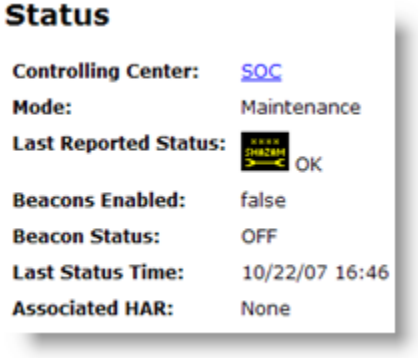


Figure 5-33 SHAZAM Details - Actions, Maint Mode

5.1.2.2.2 Status



The Status section of the SHAZAM details page is changed for R8 to show the actual beacon status (as queried from the device) when the SHAZAM model is HWG-ER02a. The existing Beacons Enabled field continues to indicate whether or not CHART is set to enable the beacons; the Beacon Status may not match this, which would be the case if there is a hardware failure. Only the Beacons Enabled field will be shown when the SHAZAM model is Viking RC2A, for CHART cannot query the Viking RC2A to determine its actual beacon status.

5.1.2.2.3 Configuration

The Configuration section of the SHAZAM details page contains many changes for R8. A new sub-section has been added to indicate the model of the SHAZAM. The Basic Configuration sub-section is changed to show the relay selection if the SHAZAM is a HWG ER02a model. The Location sub-section of the Configuration section is unchanged in R8. The Communication Settings sub-section is changed in R8 to show information pertaining to TCP/IP communications when the port type is TCP/IP (which is the only port type supported by the HWG ER02a). The Alert And Notifications section is new for R8 and applies to all SHAZAMS. It shows the currently configured operation centers that are to receive alerts for the SHAZAM and the currently configured groups that are to receive notifications.

Figure 5-34 SHAZAM Details, Configuration

5.1.2.3 Edit Model

The Edit Model form is accessed by clicking the "change" link next to the model name in the Model section of the SHAZAM details page (see below). This link is only available if the SHAZAM is offline.

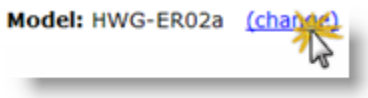


Figure 5-35 SHAZAM Change Model Link

After clicking the change link, the Change Model Type page is shown (see below). The page contains a single select box to allow the user to select the model. A warning on this page reminds the user that changing the model may require changes to the basic settings and/or communication settings. This is due to the fact that the Viking RC2A supports only Telephony communications and the HWG-ER02a supports only TCP/IP communications. When switching models, it is likely that the settings for the associated communications method were not set up previously or need changing. Additionally the HWG-ER02a has a Relay Selection setting within the basic configuration that may need to be changed if changing the model from a Viking RC2A to an HWG-ER02a.

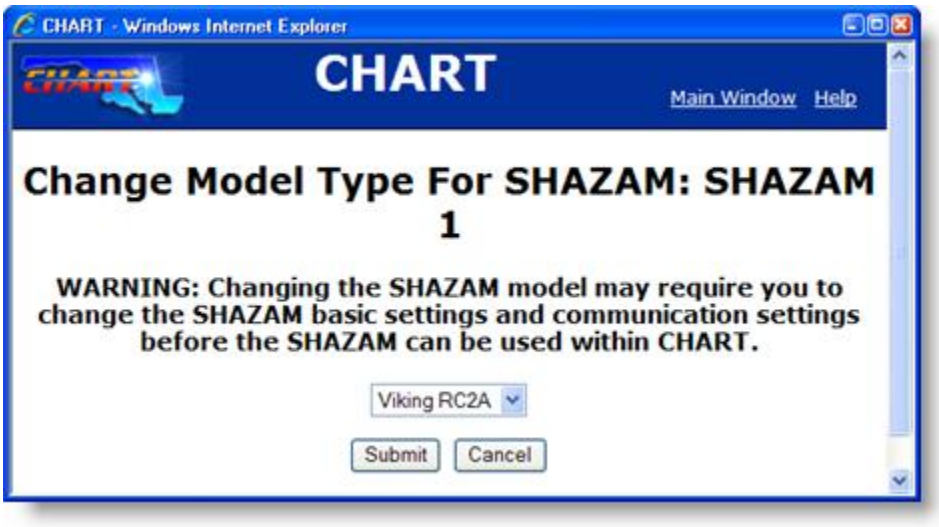


Figure 5-36 SHAZAM Change Model Type Form

5.1.2.4 Edit Basic Configuration

The Edit Basic Configuration form is accessed by clicking the Edit link next to the Basic Configuration section header on the SHAZAM details page as shown below.

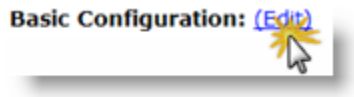


Figure 5-37 SHAZAM Edit Basic Configuration Link

After clicking the Edit link, the Edit Basic Settings form is shown (see below). This form is changed for R8 to include the model name, which is a view only field. When the model is HWG-ER02a, a new field for R8 is shown to allow the Relay to be selected.

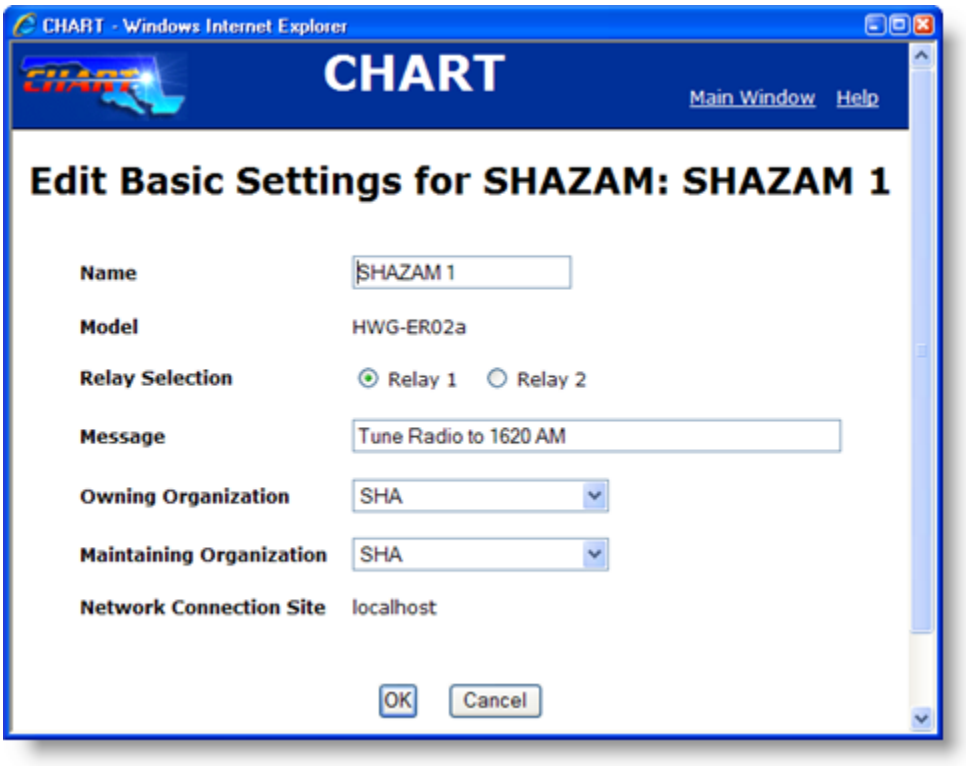
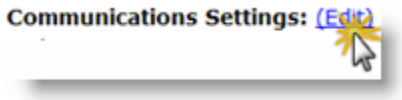


Figure 5-38 SHAZAM Edit Basic Settings Form

5.1.2.5 Edit Communications Settings

The Edit Communications Settings form is accessed by clicking the Edit link next to the Communications Settings section header on the SHAZAM details page.

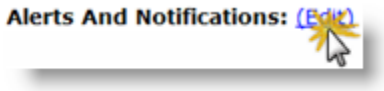


After clicking the Edit link, the Edit Comm Settings form is shown. This form is changed in R8 to support TCP/IP settings. The Port Type field is read-only and is based on the SHAZAM model. The HWG-ER02a supports only TCP/IP communications while the Viking RC2A supports only Telephony communications. When this form is accessed for a Viking RC2A SHAZAM, the form is unchanged. When accessed for a HWG-ER02a SHAZAM, the form allows the IP address and port to be set (see below). The Refresh Enabled and Refresh Interval fields apply to both models.

Figure 5-39 SHAZAM Edit Comm Settings Form

5.1.2.6 Edit Alerts and Notifications Settings

The Edit Alerts and Notifications Settings form is accessed by clicking the Edit link next to the Alerts and Notifications section heading on the SHAZAM details page.



After clicking the Edit link, the Edit Alert and Notifications Settings form is shown. The user can choose the operations center that is to receive comm failure alerts and the operations center to receive hardware failure alerts. The user can also select the group to receive comm failure notifications and the group to receive hardware failure notifications. Any of the selections can be set to "None" to disable that type of alert or notification.

Figure 5-40 SHAZAM Alert and Notification Settings Form

5.2 System Interfaces

The System Interfaces package models the CHART IDL.

5.2.1 Class Diagrams

5.2.1.1 AlertManagement (Class Diagram)

This class diagram shows the system interfaces that make the AlertManagement capability of CHART2 system.

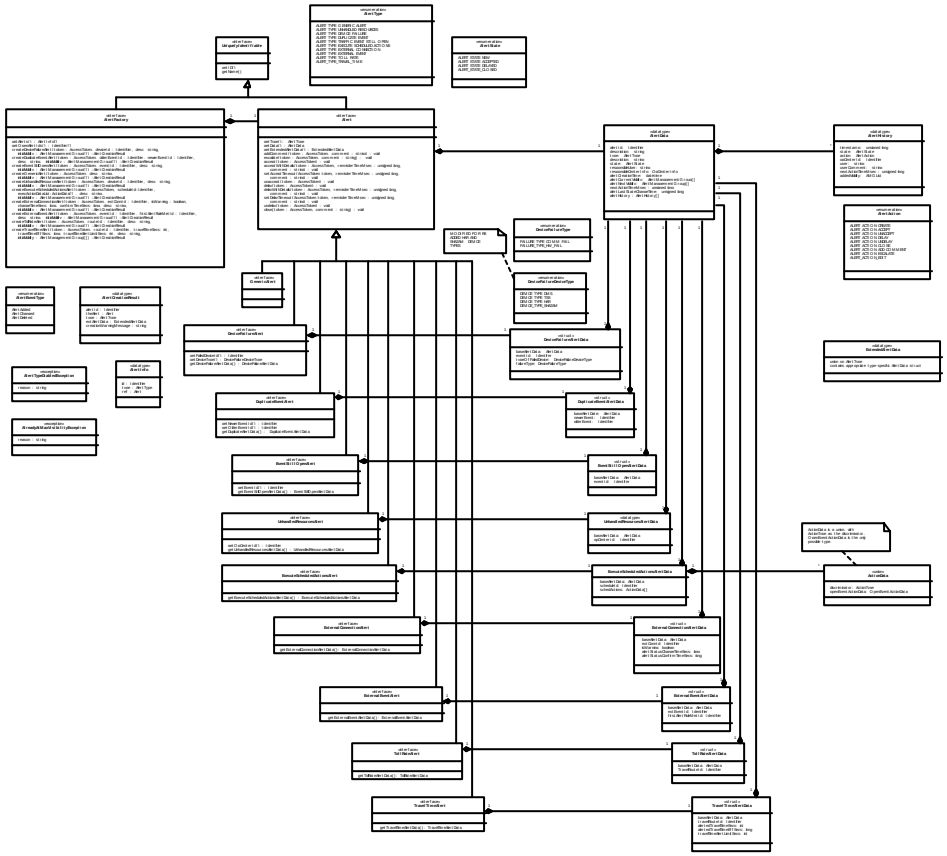


Figure 5-41. AlertManagement (Class Diagram)

5.2.1.1.1 ActionData (Class)

This IDL union holds the data used to describe a schedule action. It has been designed as a union discriminated by the enumeration ActionType to support schedule actions to be determined in future releases of CHART. Currently the only supported variant is the OpenEventAction.

5.2.1.1.2 Alert (Class)

This is a CORBA interface that provides access to information pertaining to an Alert and provides operations used to manage an alert.

5.2.1.1.3 AlertAction (Class)

This IDL enumeration defines the actions that can be done to an Alert.

5.2.1.1.4 AlertCreationResult (Class)

This IDL struct represents the data that will be returned as a result of an alert creation using the AlertFactory calls. It includes: alert id, alert CORBA reference, alert type, extended alert data, and a warning string used to describe non-fatal conditions when creating the alert.

5.2.1.1.5 AlertData (Class)

This is a CORBA struct, defined in IDL, that contains the data that applies to all alert types.

5.2.1.1.6 AlertEventType (Class)

This IDL enumeration defines the types of CORBA Events supported in the AlertModule. Its primary use is as a discriminator value used when handling AlertEvents. These can either be Alert Added, Changed, or Deleted.

5.2.1.1.7 AlertFactory (Class)

This IDL interface contains the operations available for an Alert Factory. The AlertFactory is responsible for creating alerts

and storing alert information on the alerts that it created.

5.2.1.1.8 AlertHistory (Class)

This IDL struct contains information used to describe an action being done to an alert. A collection of these structs represents the history of the alert from beginning to end.

5.2.1.1.9 AlertInfo (Class)

This IDL struct contains information about an Alert in the system. Its primary use is to be returned as part of a list of AlertInfo objects in response to an AlertFactory's getAlerts() call.

5.2.1.1.10 AlertState (Class)

AlertState is an IDL enumeration of the four defined states for an Alert.

5.2.1.1.11 AlertType (Class)

AlertType is an IDL enumeration of the five Alert types.

5.2.1.1.12 AlertTypeDiabledException (Class)

This exception is thrown by the AlertFactory create operations if the alert type being created is disabled within the system. (Server-side clients can ignore this alert; GUI-side clients may wish to display this to the user.)

5.2.1.1.13 AlreadyAtMaxVisibilityException (Class)

This exception is thrown by the Alert escalate() operation if the alert is already at maximum visibility (no additional AMGs are configured in the backup set(s) of the AMG(s) in the current visibility list). Clients may wish to try escalation after receipt of this exception (or at any time the nextVisibility array is empty), in case an administrator may have modified the backup set of AMGs in the meanwhile.

5.2.1.1.14 DeviceFailureAlert (Class)

This IDL interface contains operations specific to a Device Failure alert. This interface is implemented by classes representing DeviceFailureAlerts in the Chart2 System.

5.2.1.1.15 DeviceFailureAlertData (Class)

This is a CORBA struct, defined in IDL, that contains base alert data plus data specific to a DeviceFailureAlert. Specific to this alert is the traffic event id of the failed device event causing the alert. Also included is information on the device failure type.

5.2.1.1.16 DeviceFailureDeviceType (Class)

The DeviceFailureDeviceType is an enumeration of the possible device failure types supported in a device failure alert.

5.2.1.1.17 DeviceFailureType (Class)

This enumeration lists the possible types of device failures which can be communicated by a device failure alert.

5.2.1.1.18 DuplicateEventAlert (Class)

This IDL interface contains operations specific to a Duplicate Event alert. This interface is implemented by classes representing DuplicateEventAlertsDevice in the Chart2 System.

5.2.1.1.19 DuplicateEventAlertData (Class)

This is a CORBA struct, defined in IDL, that contains base alert data plus data specific to a DuplicateEventAlert. Specific to this alert are the event ids of the two probable duplicate traffic events.

5.2.1.1.20 EventStillOpenAlert (Class)

This IDL interface contains operations specific to a Event Still Open alert. This interface is implemented by classes representing EventStillOpenAlerts in the Chart2 System.

5.2.1.1.21 EventStillOpenAlertData (Class)

This is a CORBA struct, defined in IDL, that contain the base alert data plus data specific to an EventStillOpenAlert. Specific to this alert is the id of the traffic event that is still open.

5.2.1.1.22 ExecuteScheduledActionsAlert (Class)

This IDL interface contains operations specific to aExecute Scheduled Actions alert. This interface is implemented by classes representing ExecuteScheduledActionsAlert in the Chart2 System.

5.2.1.1.23 ExecuteScheduledActionsAlertData (Class)

This is a CORBA struct, defined in IDL, that contains the base alert data plus data specific to an ExecuteScheduledActionsAlert.

5.2.1.1.24 ExtendedAlertData (Class)

ExtendedAlertData is a union of the four type specific alert datatypes: DeviceFailureAlertData, DuplicateEventAlertData, EventStillOpenAlertData, and UnhandledResourceAlertData. Note that the GenericAlert does not include any type specific data. The AlertType enumeration is used as the discriminator over the data in this union.

5.2.1.1.25 ExternalConnectionAlert (Class)

This IDL interface contains operations specific to an External Connection Alert, which indicates trouble with a connection between CHART and an external system.

5.2.1.1.26 ExternalConnectionAlertData (Class)

This IDL structure contains data specific to an External Connection Alert, e.g., the ID of the interface which is having trouble and a flag indicating whether the connection is in failure or warning status, the timestamp it transitioned. (The GUI displays additional data which is best acquired from the GUI's object cache.) (Text in the base AlertData structure provides a textual description and alert management data.)

5.2.1.1.27 ExternalEventAlert (Class)

This IDL interface contains operations specific to an External Event Alert, which indicates an event has arrived from an external system which satisfies criteria a CHART administrator has defined to flag an external event as significant enough to warrant this alert.

5.2.1.1.28 ExternalEventAlertData (Class)

This IDL structure contains data specific to an External Event Alert, e.g., the ID of the event and the ID of the first rule found that requested an alert be sent. (Text in the base AlertData structure provides a textual description and alert management data.)

5.2.1.1.29 GenericAlert (Class)

This IDL interface contains operations specific to a Generic alert. This interface is implemented by classes representing GenericAlerts in the Chart2 System.

5.2.1.1.30 TollRateAlert (Class)

This IDL interface contains operations specific to an Toll Rate Alert, which indicates a travel route which had a currently active toll rate no longer does in a more recently received toll rate update document from a toll rate provider. (This alert is not sent if a toll rate expires due to an absence of any current toll rate document -- such an event would have triggered one external connection alert and does not need to also trigger a multitude of individual toll rate alerts as well.)

5.2.1.1.31 TollRateAlertData (Class)

This IDL structure contain data specific to a Toll Rate Alert, e.g., the travel route which no longer has data for its toll rate. (Text in the base AlertData structure provides a textual description and alert management data.)

5.2.1.1.32 TravelTimeAlert (Class)

This IDL interface contains operations specific to an Travel Time Alert, which indicates the travel time associated with a travel route is high enough to warrant this alert.

5.2.1.1.33 TravelTimeAlertData (Class)

This IDL structure contains data specific to a Travel Time Alert, e.g., the travel time limit and the travel time which exceeded the limit. (Text in the base AlertData structure provides a textual description and alert management data.)

5.2.1.1.34 UnhandledResourcesAlert (Class)

This IDL interface contains operations specific to a Unhandled Resources alert. This interface is implemented by classes representing UnhandledResourceAlerts in the Chart2 System.

5.2.1.1.35 UnhandledResourcesAlertData (Class)

This is a CORBA struct, defined in IDL, that contains the base alert data plus data specific to an UnhandledResourcesAlert.

5.2.1.1.36 UniquelyIdentifiable (Class)

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

5.2.1.2 HARControl (Class Diagram)

This class diagram contains the interfaces used relating to the control of Highway Advisory Radio (HAR).

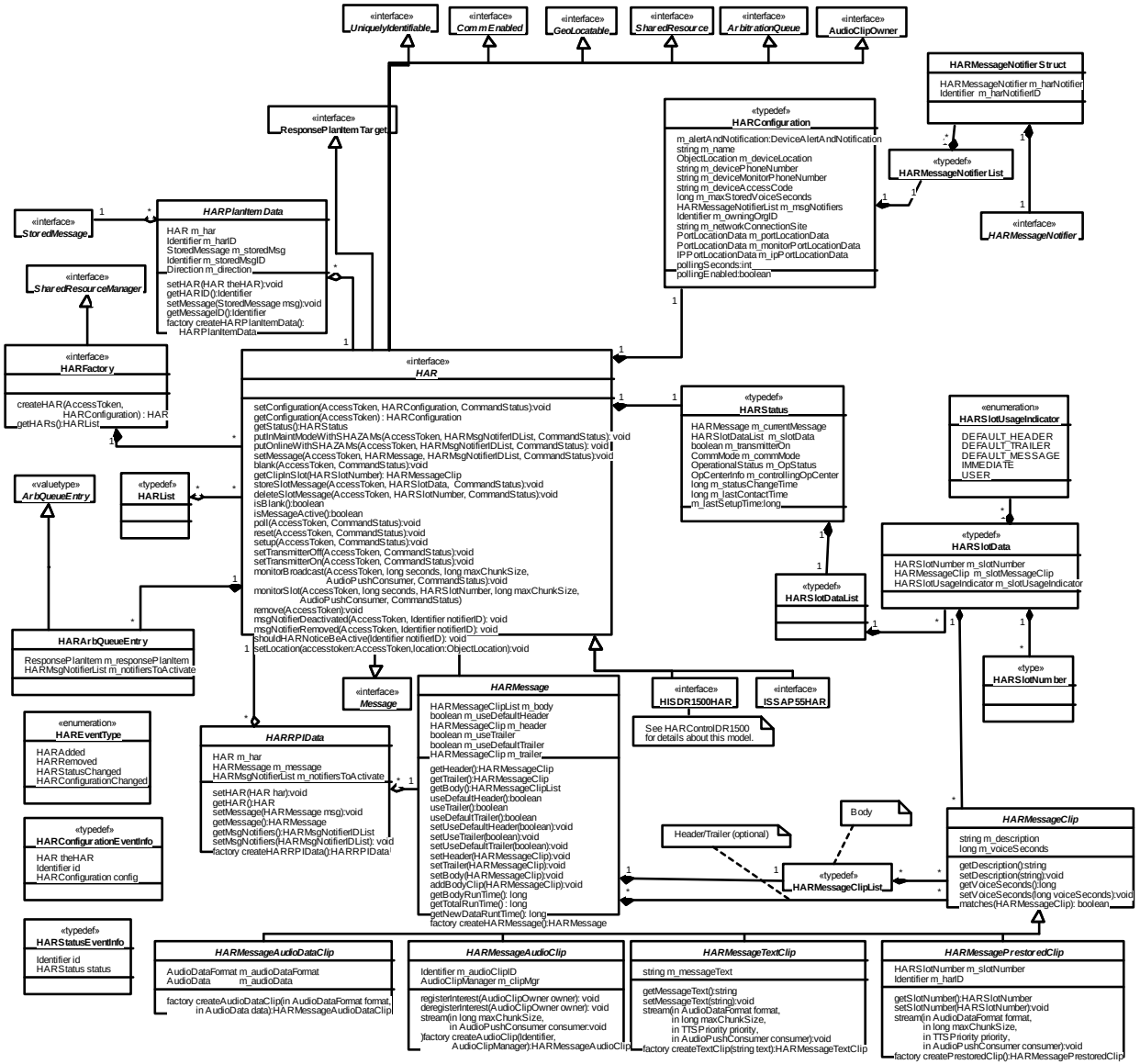


Figure 5-42. HARControl (Class Diagram)

5.2.1.2.1 ArbitrationQueue (Class)

An ArbitrationQueue is a queue that arbitrates the usage of a device. The evaluation of the queue determines which message(s) should be on the device, based upon the priority of the queue entries. When entries are added to the queue, they are assigned a priority level based on the type of traffic event with which they are associated, and also upon the current contents of the queue. The priority of the queue entries can be modified after they are added to the queue. The queue is evaluated when the device is online and queue entries are added or removed, when an entry's priority is modified, or when the device is put online.

5.2.1.2.2 ArbQueueEntry (Class)

This class is used for an entry on the arbitration queue, for a single message, and for a single traffic event. (It is possible, in the case of HARNotifierArbQueueEntry objects, that certain ArbQueueEntries can be on behalf of multiple TrafficEvents. In such cases, one TrafficEvent among all those involved is picked to be the responsible TrafficEvent stored in m_indicator, the ArbQueueEntryIndicator for the entry.)

5.2.1.2.3 AudioClipOwner (Class)

This interface allows the AudioClipManager to check whether there are any parties interested in an audio clip. If no AudioClipOwners claim interest in a clip, the clip can be deleted.

5.2.1.2.4 CommEnabled (Class)

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of device are allowed to help in troubleshooting.

5.2.1.2.5 GeoLocatable (Class)

This interface is implemented by objects that can provide location information to their users.

5.2.1.2.6 HAR (Class)

This class is used to represent a Highway Advisory Radio (HAR) device. A HAR is used to broadcast traffic related information over a localized radio transmitter, making the information available to the traveler. This interface contains methods for getting and setting configuration, getting status, changing communications modes of a HAR, and manipulating and monitoring the HAR in maintenance and online modes.

5.2.1.2.7 HARArbQueueEntry (Class)

This class is an arbitration queue entry used to set the message on a HAR on behalf of a traffic event. This entry also specifies the HARMessageNotifiers to be activated when the message is activated.

5.2.1.2.8 HARConfiguration (Class)

This class (struct) contains configuration data for a HAR device. It is used to transmit current configuration data from the HAR to the client, and to transmit proposed new configuration data from the client to the HAR. It is also used internally by the HARService to maintain its configuration in memory, and is used to transmit configuration data to/from the HAR to the HARControlDB database interface class. Device Location member has been modified for R3B3. Now it contains a detailed location information.

5.2.1.2.9 HARConfigurationEventInfo (Class)

This class defines data (HARConfiguration, and HAR ID and reference) pushed with a HARConfigurationChanged and HARAdded CORBA event.

5.2.1.2.10 HAREventType (Class)

This enumeration defines the types of CORBA events that are pushed on a HARControl event channel.

5.2.1.2.11 HARFactory (Class)

This CORBA interface allows new HAR objects to be added to the system. It also allows a requester to acquire a list of HAR objects under the domain of the specific HARFactory object.

5.2.1.2.12 HARList (Class)

The HARList class is a collection of HAR objects.

5.2.1.2.13 HARMessage (Class)

This utility class represents a message which is capable of being stored on a HAR. It stores the HAR message as a HAR message header, body and footer. The HARMessage can be configured to use the default header or can provide a custom header clip. The trailer can be specified to use the default trailer, or no trailer, or a custom trailer clip can be provided. The body can consist of one or more body clips. Users must specify one and only one body clip, but the HAR Service can combine messages for broadcast as a single combined message on a HAR, up to a maximum run length.

5.2.1.2.14 HARMessageAudioClip (Class)

This class is a thin wrapper for recorded voice that is to be played on a HAR. This class is passed around the system, wherever possible instead of passing the actual voice data contained in the initial HARMessageAudioDataClip. When the actual voice data is needed to play to the user or to program the HAR device, this object's streamer is used to stream the actual voice data back to an AudioPushConsumer specified by the requester.

5.2.1.2.15 HARMessageAudioDataClip (Class)

This class is a message clip that contains audio data and the format of the audio data. Because audio data can be very large, this type of clip is reserved for use when recorded voice is first entered into the system. Recorded voice that already exists in the system is passed throughout the system using HARMessageAudioClip to avoid sending the large audio data when possible. A HARMessageAudioClip can stream the associated data back to an audio consumer when needed, by contacting its AudioClipManager.

5.2.1.2.16 HARMessageClip (Class)

This class represents a section of a HAR message. A HARMessage typically contains one to three clips: a body plus an optional header and optional trailer. A combined HARMessage which is stored on (broadcast from) a HAR can one or more clips, an optional header, optional trailer, and one or more body clips. See HARMessage for details. A HARMessageClip can be either plain text which would need to be converted to audio prior to broadcast, or audio (WAV) format, or it can refer to a clip which is prestored in a specific target HAR already. Audio clips are normally passed around as lightweight HARMessageAudioClips, which are created from HARMessageAudioDataClips typically at the point where the HARMessageAudioClip first enters a server.

5.2.1.2.17 HARMessageClipList (Class)

The HARMessageClipList is a collection of HARMessageClip objects. It is used to specify multiple clips contained in the

body of a HARMessage. While a HARMessage specified by a user can contain only one body clip, a HARMessage generated by the HAR Service can contain multiple body clips, as a result of combining more than one message into a single message for download to and broadcast by a HAR.

5.2.1.2.18 HARMessageNotifier (Class)

The HARMessageNotifier class specifies an interface to be implemented by devices that can be used to notify the traveler to tune in to a radio station to hear a traffic message being broadcast by a HAR. A HARMessageNotifier is directional and allows users of the device to better determine if activation of the device is warranted for the message being broadcast by the HAR. This interface can be implemented by SHAZAM devices and by DMS devices which are allowed to provide a SHAZAM-like message.

5.2.1.2.19 HARMessageNotifierList (Class)

This class defines a list of HARMessageNotifierStruct objects.

5.2.1.2.20 HARMessageNotifierStruct (Class)

This class (struct) defines structure used for specifying a HARMessageNotifier, containing the notifier's ID and reference.

5.2.1.2.21 HARMessagePrestoredClip (Class)

This class stores data used to identify the usage of a clip that has already been stored on a specific HAR device.

5.2.1.2.22 HARMessageTextClip (Class)

This class represents a HAR message content object which is in plain text format. This message can be checked for banned words and will be converted into a voice message using a speech engine, for downloading to a HAR device or to preview the voice audio to a user.

5.2.1.2.23 HARPlanItemData (Class)

This class is used to associate a message with a HAR for use in Plans.

5.2.1.2.24 HARRPIData (Class)

This class represents an item in a traffic event response plan that is capable of issuing a command to put a message on a HAR

when executed. When the item is executed, it adds an ArbQueueEntry to the specified HAR, which stores the entry in its MessageQueue. When the item's execution is revoked, or the item is removed from the response plan (manually or implicitly through closing the traffic event) the item asks the HAR to remove the entry. The HARRPIData object also allows specification of a subset (0 to all) of the HARNotifier devices (SHAZAM or DMS devices acting as SHAZAMs) to be activated if and while the message is being broadcast on the HAR.

5.2.1.2.25 HARSlotData (Class)

This struct defines the data used to identify the contents and usage of a slot in the HAR controller.

5.2.1.2.26 HARSlotDataList (Class)

The HARSlotDataList class is simply a collection of HARSlotData objects.

5.2.1.2.27 HARSlotNumber (Class)

The HARSlotNumber is an integer used to specify slot numbers on a HAR controller.

5.2.1.2.28 HARSlotUsageIndicator (Class)

This enum defines indicators used to show the usage of a given slot in the HAR controller.

5.2.1.2.29 HARStatus (Class)

This class (struct) contains data that indicates the current status of a HAR device. The data contained in this class is that status information which can be transmitted from the HAR to the client as necessary. This struct is also used to within the HAR Service to transmit data to/from the HARControlDB database interface class. (The HAR implementation also contains other private status data elements which are not elements of this class.)

5.2.1.2.30 HARStatusEventInfo (Class)

This class contains data (HARStatus) that is pushed when the HARStatusChanged CORBA event is pushed on the HARControl event channel.

5.2.1.2.31 HISDR1500HAR (Class)

This interface is implemented by objects that provide for the control of an HIS model DR1500 HAR.

5.2.1.2.32 ISSAP55HAR (Class)

This interface is implemented by objects that provide for the control of an ISS model AP55 HAR.

5.2.1.2.33 Message (Class)

This class represents a message that will be used while activating devices. This class provides a means to check if the message contains any banned words given a Dictionary object. Derived classes extend this class to provide device specific message data.

5.2.1.2.34 ResponsePlanItemTarget (Class)

This interface represents an object that can be a target of a response plan item.

5.2.1.2.35 SharedResource (Class)

The SharedResource interface is implemented by any object that may have an operations center responsible for the disposition of the resource while the resource is in use.

5.2.1.2.36 SharedResourceManager (Class)

The SharedResourceManager interface is implemented by classes that manage shared resources. Implementing classes must be able to provide a list of all shared resources under their management. Implementing classes must also be able to tell others if there are any resources under its management that are controlled by a given operations center. The shared resource manager is also responsible for periodically monitoring its shared resources to detect if the operations center controlling a resource doesn't have at least one user logged into the system. When this condition is detected, the shared resource manager must push an event on the ResourceManagement event channel to notify others of this condition.

5.2.1.2.37 StoredMessage (Class)

This class holds a message object that is stored in a message in a library. It contains attributes such as category and message description which are used to allow the user to organize messages.

5.2.1.2.38 UniquelyIdentifiable (Class)

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

5.2.1.3 DeviceManagement (Class Diagram)

This class diagram shows device interfaces that are common among devices.

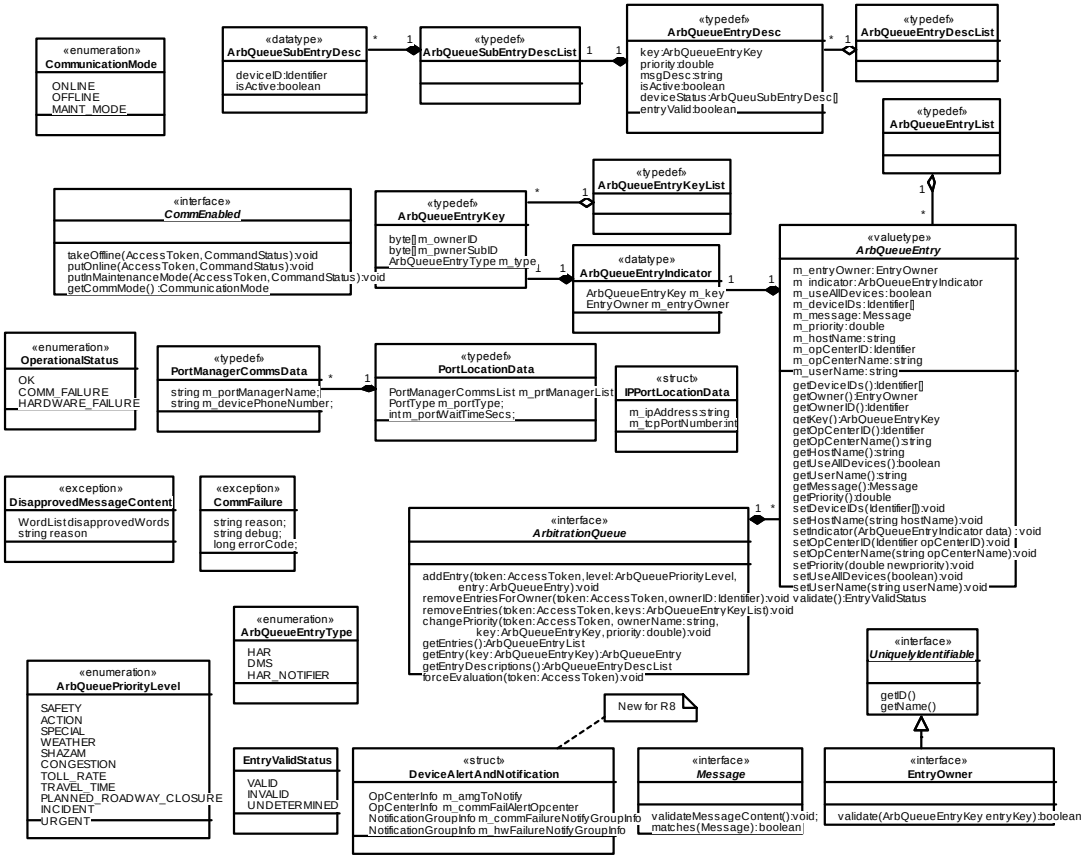


Figure 5-43. DeviceManagement (Class Diagram)

5.2.1.3.1 ArbitrationQueue (Class)

An ArbitrationQueue is a queue that arbitrates the usage of a device. The evaluation of the queue determines which message(s) should be on the device, based upon the priority of the queue entries. When entries are added to the queue, they are assigned a priority level based on the type of traffic event with which they are associated, and also upon the current contents of the queue. The priority of the queue entries can be modified after they are added to the queue. The queue is evaluated when the device is online and queue entries are added or removed, when an entry's priority is modified, or when the device is put online.

5.2.1.3.2 ArbQueueEntry (Class)

This class is used for an entry on the arbitration queue, for a single message, and for a single traffic event. (It is possible, in the case of HARNotifierArbQueueEntry objects, that certain ArbQueueEntries can be on behalf of multiple TrafficEvents. In such cases, one TrafficEvent among all those involved is picked to be the responsible TrafficEvent stored in m_indicator, the ArbQueueEntryIndicator for the entry.)

5.2.1.3.3 ArbQueueEntryDesc (Class)

This structure is used to provide a description of an entry on the arbitration queue.

5.2.1.3.4 ArbQueueEntryDescList (Class)

Collection of ArbQueueEntryDesc objects.

5.2.1.3.5 ArbQueueEntryIndicator (Class)

The ArbQueueEntryIndicator contains data necessary to specify a unique ArbQueueEntry object; in addition, it contains a reference to the TrafficEvent which is responsible for the entry.

5.2.1.3.6 ArbQueueEntryKey (Class)

This class contains the Traffic Event ID and RPI ID and is used to identify a specific ArbQueueEntry. In some cases (e.g., for HARNotifierArbQueueEntry objects), the RPI ID is the string representing a null Identifier.

5.2.1.3.7 ArbQueueEntryKeyList (Class)

A collection of ArbQueueEntryKey objects.

5.2.1.3.8 ArbQueueEntryList (Class)

A collection of ArbQueueEntry objects.

5.2.1.3.9 ArbQueueEntryType (Class)

Enumeration of all possible types of entries that could be on an arbitration queue.

5.2.1.3.10 ArbQueuePriorityLevel (Class)

Enumeration of all possible priority levels of the arbitration queue. All entries in the queue fit into one of these levels. The levels are named after the types of messages that are typically mapped into them. However, any message can exist in any level and new types of messages could be mapped into these levels. Thus, the levels could have been named, LOWEST through HIGHEST. The names chosen have been used to provide some indication of the likely usage of the levels. TOLL_RATE and TRAVEL_TIME levels have been added for R3B3.

5.2.1.3.11 ArbQueueSubEntryDesc (Class)

This structure hold ArbQueueEntry "device-level detail for one "sub-device (such as a constituent HAR within a SyncHAR). It holds the ID of the device and an indication as to whether the entry is active for this particular subdevice. An ArbQueueEntry for a conglomerate device (such as a SyncHAR) will contain a list of these structures, one for each constituent HAR the entry is destined for.

5.2.1.3.12 ArbQueueSubEntryDescList (Class)

This is an array of ArbQueueSubEntryDesc. It holds a list of "sub-devices" (such as constituent HARs of a SyncHAR) for which an arb queue entry is destined, and for which of those devices the entry is active.

5.2.1.3.13 CommEnabled (Class)

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of device are allowed to help in troubleshooting.

5.2.1.3.14 CommFailure (Class)

This exception is to be thrown when an error is detected connecting to or communicating with a device.

5.2.1.3.15 CommunicationMode (Class)

The CommunicationMode class enumerates the modes of operation for a device: ONLINE, OFFLINE, and MAINT_MODE. ONLINE is used to indicate the device is available to the operational system. OFFLINE is used to indicate the device is not available to the online system and communications to the device have been disabled. MAINT_MODE is used to indicate that the device is available only for maintenance / repair activities and testing.

5.2.1.3.16 DeviceAlertAndNotification (Class)

This structure stores device alert and notification data.

5.2.1.3.17 DisapprovedMessageContent (Class)

This exception is thrown when a text message to be put on a device contains words that are not approved. This exception is also thrown if an attempt is made to put the device in an invalid display state, such as putting the Beacons ON for a blank DMS.

5.2.1.3.18 EntryOwner (Class)

Interface which must be implemented by any class which is responsible for putting an ArbQueueEntry on a device's arbitration queue. This validate method of this interface can be called by the device to determine continued validity of the entry (either during recovery or as a final check of the validity of an entry before putting its message on the device).

5.2.1.3.19 EntryValidStatus (Class)

This enumeration is used to track whether an arb queue entry has been validated by its EntryOwner (interface). The possible values are VALID, INVALID, and UNDETERMINED.

5.2.1.3.20 IPPortLocationData (Class)

this structure defines the connection information of a tcp/ip port.

5.2.1.3.21 Message (Class)

This class represents a message that will be used while activating devices. This class provides a means to check if the message contains any banned words given a Dictionary object. Derived classes extend this class to provide device specific message data.

5.2.1.3.22 OperationalStatus (Class)

The OperationalStatus class enumerates the types of operational status a device can have: OK (normal mode), COMM_FAILURE (no communications to the device), or HARDWARE_FAILURE (device is reachable but is reporting a hardware failure).

5.2.1.3.23 PortLocationData (Class)

This class contains configuration data that specifies the communication server(s) to use to communicate with a device.

m_commsData - One or more objects identifying the communications server (PortManager) to use to communicate with the device, in order of preference.

m_portType - The type of port to use to communicate with the device (ISDN modem, POTS modem, direct, etc.)

m_portWaitTimeSecs - The maximum number of seconds to wait when attempting to acquire a port from a port manager.

5.2.1.3.24 PortManagerCommsData (Class)

This class contains values that identify a port manager and the phone number to dial to access a device from the given port manager. This class exists to allow for the phone number used to access a device to differ based on the port manager to take into account the physical location of the port manager within the telephone network. For example, when dialing a device from one location the call may be long distance but when dialing from another location the call may be local.

5.2.1.3.25 UniquelyIdentifiable (Class)

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

5.2.1.4 HARControlDR1500 (Class Diagram)

This class diagram contains the interfaces used relating to the control of DR1500 Highway Advisory Radio (HAR).

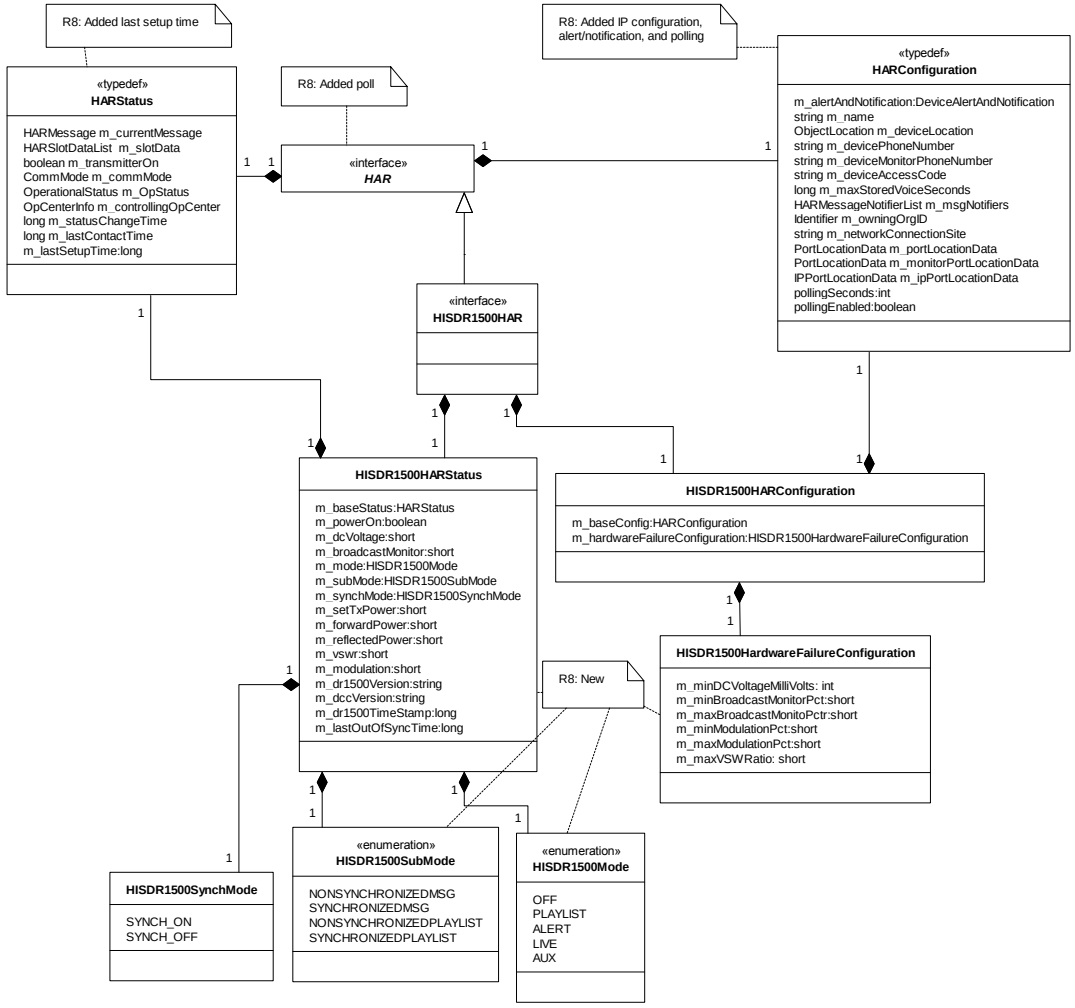


Figure 5-44. HARControlDR1500 (Class Diagram)

5.2.1.4.1 HAR (Class)

This class is used to represent a Highway Advisory Radio (HAR) device. A HAR is used to broadcast traffic related information over a localized radio transmitter, making the information available to the traveler. This interface contains methods for getting and setting configuration, getting status, changing communications modes of a HAR, and manipulating and monitoring the HAR in maintenance and online modes.

5.2.1.4.2 HARConfiguration (Class)

This class (struct) contains configuration data for a HAR device. It is used to transmit current configuration data from the HAR to the client, and to transmit proposed new configuration data from the client to the HAR. It is also used internally by the HARService to maintain its configuration in memory, and is used to transmit configuration data to/from the HAR to the HARControlDB database interface class. Device Location member has been modified for R3B3. Now it contains a detailed location information.

5.2.1.4.3 HARStatus (Class)

This class (struct) contains data that indicates the current status of a HAR device. The data contained in this class is that status information which can be transmitted from the HAR to the client as necessary. This struct is also used to within the HAR Service to transmit data to/from the HARControlDB database interface class. (The HAR implementation also contains other private status data elements which are not elements of this class.)

5.2.1.4.4 HISDR1500HAR (Class)

This interface is implemented by objects that provide for the control of an HIS model DR1500 HAR.

5.2.1.4.5 HISDR1500HARConfiguration (Class)

This type contains DR1500 specific configuration fields. In R8 this type is modified to contain fields for polling.

5.2.1.4.6 HISDR1500HardwareFailureConfiguration (Class)

This structure defines DR1500 HAR and transmitter status thresholds to determine when a hardware failure alert/notification should be sent.

5.2.1.4.7 HISDR1500HARStatus (Class)

This type contains statis fields specific to DR1500 HARs

5.2.1.4.8 HISDR1500Mode (Class)

This enumeration contains DR1500 mode values.

5.2.1.4.9 HISDR1500SubMode (Class)

This enumeration contains DR1500 sub mode values.

5.2.1.4.10 HISDR1500SynchMode (Class)

This structure is an indicator for whether a DR1500 HAR with a synchronized playlist is in fact operating in synchronized mode.

5.2.1.5 HARNotification (Class Diagram)

This Class Diagram shows the classes involved in manipulating HAR message notifications. The HAR notifiers can be SHAZAMs or DMS devices that are acting as SHAZAMs. Note that R1B2 prevents a DMS SHAZAM message from overwriting another type of DMS message.

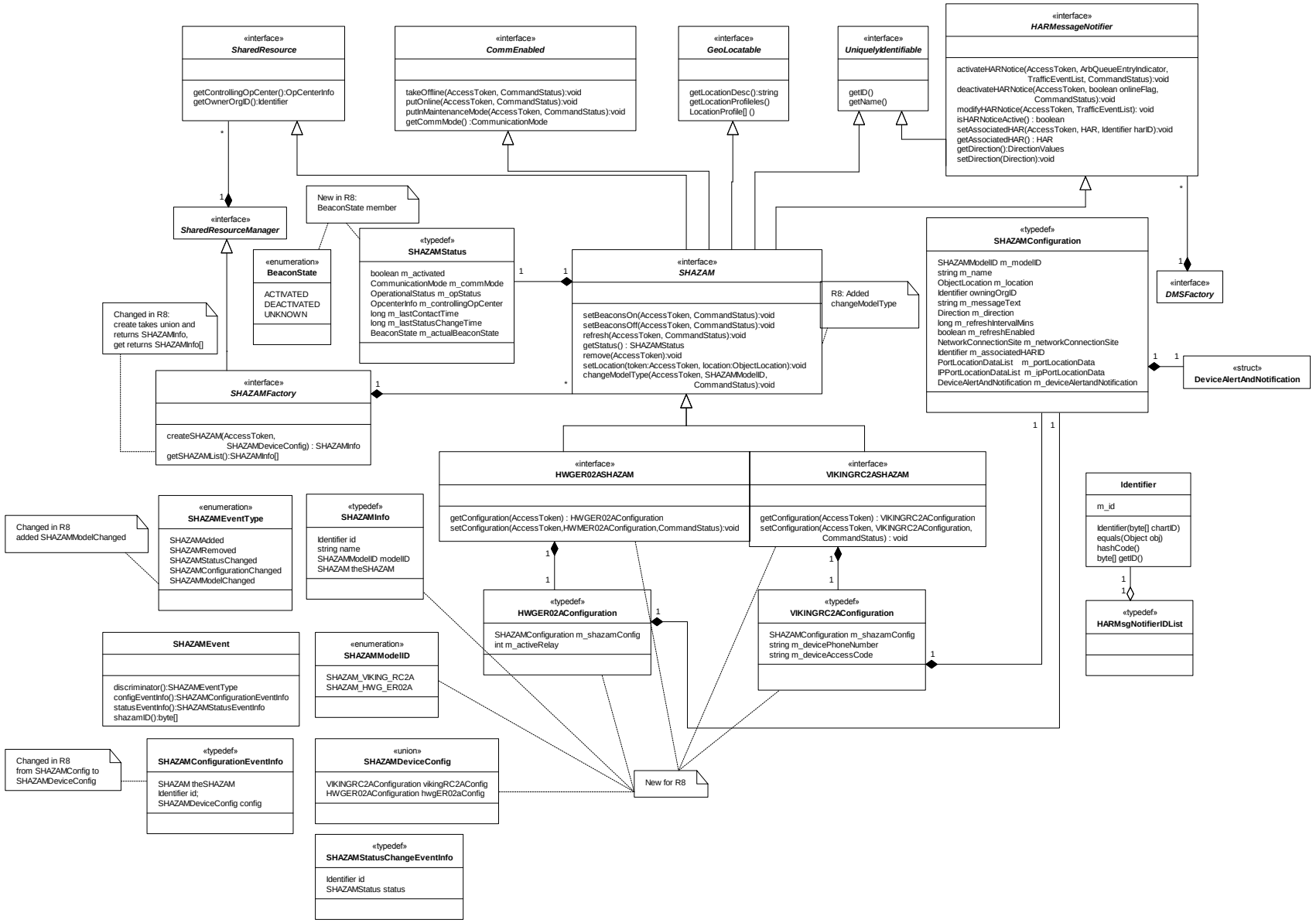


Figure 5-45. HARNotification (Class Diagram)

5.2.1.5.1 BeaconState (Class)

The valid values for the current beacon state, as queried from the device.

5.2.1.5.2 CommEnabled (Class)

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of device are allowed to help in troubleshooting.

5.2.1.5.3 DeviceAlertAndNotification (Class)

This structure stores device alert and notification data.

5.2.1.5.4 DMSFactory (Class)

The DMSFactory class specifies the interface to be used to create DMS objects within the Chart II system. It also provides a method to get a list of DMS devices currently in the system.

5.2.1.5.5 GeoLocatable (Class)

This interface is implemented by objects that can provide location information to their users.

5.2.1.5.6 HARMessageNotifier (Class)

The HARMessageNotifier class specifies an interface to be implemented by devices that can be used to notify the traveler to tune in to a radio station to hear a traffic message being broadcast by a HAR. A HARMessageNotifier is directional and allows users of the device to better determine if activation of the device is warranted for the message being broadcast by the HAR. This interface can be implemented by SHAZAM devices and by DMS devices which are allowed to provide a SHAZAM-like message.

5.2.1.5.7 HARMsgNotifierIDList (Class)

This typedef is a sequence of HARMessageNotifier identifiers.

5.2.1.5.8 HWGER02AConfiguration (Class)

This class contains SHAZAMConfiguration plus data that is specific to a HWG ER02a SHAZAM.

5.2.1.5.9 HWGER02ASHAZAM (Class)

This interface is used to provide access to configuration data specific to the HWG ER02a SHAZAM.

5.2.1.5.10 Identifier (Class)

Wrapper class for a CHART2 identifier byte sequence. This class will be used to add identifiable objects to hash tables and perform subsequent lookup operations.

5.2.1.5.11 SharedResource (Class)

The SharedResource interface is implemented by any object that may have an operations center responsible for the disposition of the resource while the resource is in use.

5.2.1.5.12 SharedResourceManager (Class)

The SharedResourceManager interface is implemented by classes that manage shared resources. Implementing classes must be able to provide a list of all shared resources under their management. Implementing classes must also be able to tell others if there are any resources under its management that are controlled by a given operations center. The shared resource manager is also responsible for periodically monitoring its shared resources to detect if the operations center controlling a resource doesn't have at least one user logged into the system. When this condition is detected, the shared resource manager must push an event on the ResourceManagement event channel to notify others of this condition.

5.2.1.5.13 SHAZAM (Class)

This interface class is used to identify the common SHAZAM-specific methods which can be used to interface with a SHAZAM field device. It specifies methods for activating and deactivating the SHAZAM in maintenance mode, refreshing the SHAZAM (commanding the device to its last known status) and removing the SHAZAM. This interface is implemented by a SHAZAMImpl class, which uses a helper ProtocolHdlr class to perform the model specific protocol for device command

and control.

5.2.1.5.14 SHAZAMConfiguration (Class)

This class contains data that specifies the configuration of a SHAZAM device. It is used to communicate configuration information to/from the database, and to/from the GUI clients. The GUI sends a SHAZAMConfiguration when creating a SHAZAM or modifying the configuration of an existing SHAZAM.Device Location member has been modified for R3B3. Now it contains a detailed location information.

5.2.1.5.15 SHAZAMConfigurationEventInfo (Class)

This class contains data (a SHAZAMDeviceConfig object) that is pushed on the SHAZAMControl CORBA event channel with a SHAZAMConfigurationChanged or SHAZAMAdded event type.

5.2.1.5.16 SHAZAMDeviceConfig (Class)

This structure stores configuration information for a SHAZAM that is specific to the SHAZAM model.

5.2.1.5.17 SHAZAMEvent (Class)

This class is a CORBA union that is used when pushing SHAZAM related events. The type of data included in it is dependent upon the type of event being pushed as follows:

SHAZAMAdded: SHAZAMConfigurationEventInfo SHAZAMConfigurationChanged: SHAZAMConfigurationEventInfo SHAZAMStatusChanged: SHAZAMStatusChangeEventInfo SHAZAMRemoved: Identifier (SHAZAM ID) SHAZAMModelChanged: SHAZAMConfigurationEventInfo (new for R8)

5.2.1.5.18 SHAZAMEventType (Class)

This enum defines the types of CORBA events that are pushed on a SHAZAM control event channel.

5.2.1.5.19 SHAZAMFactory (Class)

The SHAZAMFactory class specifies the interface to be used to create SHAZAM objects within the Chart II system. It also provides a method to get a list of SHAZAM devices currently in the system.

5.2.1.5.20 SHAZAMInfo (Class)

This struct contains information about a single SHAZAM.

5.2.1.5.21 SHAZAMModelID (Class)

This enum contains the supported SHAZAM hardware models.

5.2.1.5.22 SHAZAMStatus (Class)

This class contains the current status of a SHAZAM device. This class is used to store status within the SHAZAM object, and is also used to communicate configuration information to/from the database, and to the GUI clients (one-way).

5.2.1.5.23 SHAZAMStatusChangeEventInfo (Class)

This class contains data (a SHAZAMStatus object) that is pushed on a SHAZAMControl event channel with a SHAZAMStatusChanged event.

5.2.1.5.24 UniquelyIdentifiable (Class)

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

5.2.1.5.25 VIKINGRC2AConfiguration (Class)

This class contains SHAZAMConfiguration plus data that is specific to a Viking RC2A SHAZAM.

5.2.1.5.26 VIKINGRC2ASHAZAM (Class)

This interface is used to provide access to configuration data specific to the Viking RC2A SHAZAM.

5.2.2 Sequence Diagrams

Sequence diagrams do not apply to the System Interfaces.

5.3 Audio Common

5.3.1 Class Diagrams

5.3.1.1 AudioCommonClasses (Class Diagram)

This diagram shows classes that are commonly used for audio processing.

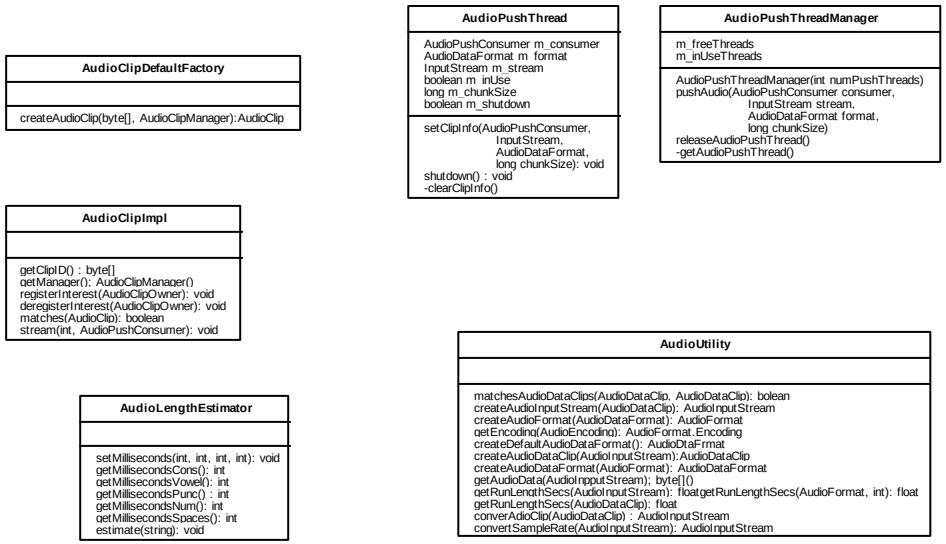


Figure 5-46. AudioCommonClasses (Class Diagram)

5.3.1.1.1 AudioClipDefaultFactory (Class)

The AudioClipDefaultFactory is the implementation of the AudioClipValueFactory used by the ORB to create AudioClipImpl objects representing the AudioClip value type.

5.3.1.1.2 AudioClipImpl (Class)

This class contains the implementation of the AudioClip value type abstract class.

5.3.1.1.3 AudioLengthEstimator (Class)

This class provides the ability to estimate the number of milliseconds it will take to play a text message based on a input parameters of N milliseconds for each consanant, vowel, punctuation, and number.

5.3.1.1.4 AudioPushThread (Class)

This class is a thread which is used to push audio clip information to an AudioPushConsumer.

5.3.1.1.5 AudioPushThreadManager (Class)

This class maintains a pool of reusable AudioPushThread objects, which can be used to push audio clip information back to the client. It provides the functionality to manage access to the AudioPushThreads.

5.3.1.1.6 AudioUtility (Class)

This class provides helper methods to use Audio related objects. It provides many static methods that aid in the conversion between java sound objects and CHART2 CORBA objects.

5.3.2 Sequence Diagrams

5.3.2.1 AudioCommonClasses:ConvertAudioClip (Sequence Diagram)

This diagram shows how audio data is converted from U-Law 8 KHz 8 bit mono to PCM 11Kh 8 bit mono format. When a request is made to convert audio Convert Audio function get source Stream in U-law format. Then we are constructing the desired format of the audio data (Encoding.PCM_UNSIGNED, Sample Rate 8 KHz, mono. Because of Sun implementation of Java Sound API does not support conversion and re-sampling we use Tritonus (open source java sound implementation). After desire format constructed we call UlawFormatConversionProvider to convert audio stream from u-law to pcm. Second step is sample rate conversion from 8Kh to 11. When target Audio Format constructed we call Audio System to re-sample source Stream with new format.

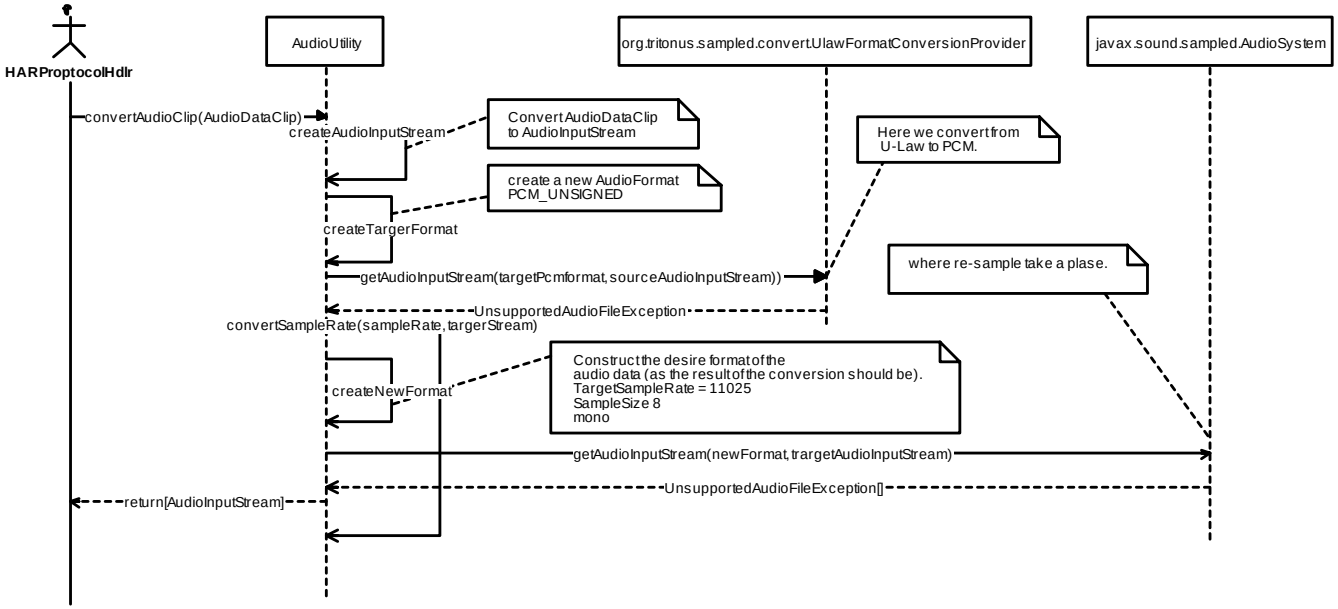


Figure 5-47 AudioCommonClasses:ConvertAudioClip (Sequence Diagram)

5.4 Device Utility

5.4.1 Class Diagrams

5.4.1.1 DeviceUtility (Class Diagram)

This class diagram shows utility classes that are useful for tasks in performing device control.

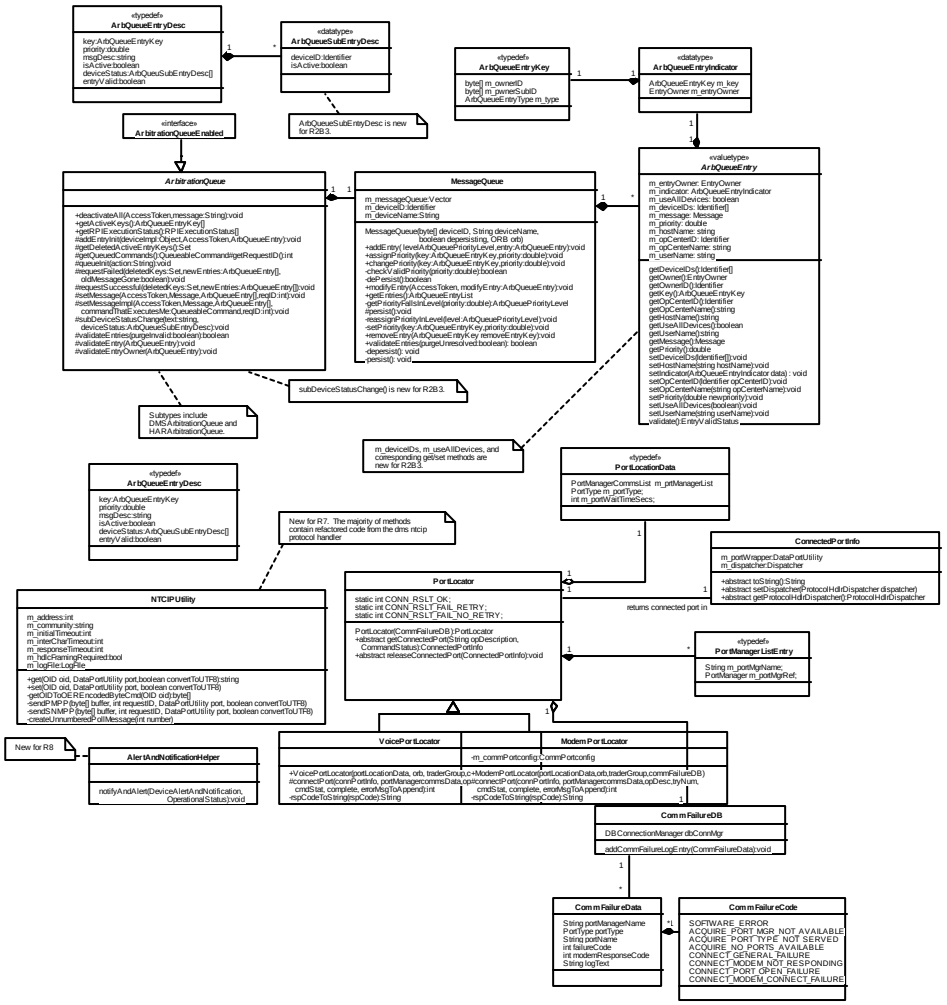


Figure 5-48. DeviceUtility (Class Diagram)

5.4.1.1.1 AlertAndNotificationHelper (Class)

This helper class provides method to sends an alert and notification messages to a notification group.

5.4.1.1.2 ArbitrationQueue (Class)

This is an abstract implementation of a generic device arbitration queue. It basically implements of the ArbitrationQueue CORBA interface (shown as ArbitrationQueueEnabled in this design). However, the official implmenters of ArbitrationQueue (ArbitrationQueueEnabled) interface are the devices themselves, CHART2DMSImpl and HARImpl. All ArbitrationQueue types of operations are delegated to an instance of this ArbitrationQueue class (one per physical device, i.e., one per instance of a device Impl class). There are device-specific concrete extensions of ArbitrationQueue for DMS and HAR, namely, DMSArbitrationQueue and HARArbitrationQueue. These provide device-specific variation.

5.4.1.1.3 ArbitrationQueueEnabled (Class)

(This interface, defined in the design in SystemInterfaces/DeviceManagement, is called ArbitrationQueue in the code, but cannot be called ArbitrationQueue in the design because there is also an ArbitrationQueue abstract class.) An ArbitrationQueue is a queue that arbitrates the usage of a device. The evaluation of the queue determines which message(s) should be on the device, based upon the priority of the queue entries. When entries are added to the queue, they are assigned a priority level based on the type of traffic event with which they are associated, and also upon the current contents of the queue. The priority of the queue entries can be modified after they are added to the queue. The queue is evaluated when the device is online and queue entries are added or removed, when an entry's priority is modified, or when the device is put online.

5.4.1.1.4 ArbQueueEntry (Class)

This class is used for an entry on the arbitration queue, for a single message, and for a single traffic event. (It is possible, in the case of HARNotifierArbQueueEntry objects, that certain ArbQueueEntries can be on behalf of multiple TrafficEvents. In such cases, one TrafficEvent among all those involved is picked to be the responsible TrafficEvent stored in m_indicator, the ArbQueueEntryIndicator for the entry.)

5.4.1.1.5 ArbQueueEntryDesc (Class)

This structure is used to provide a description of an entry on the arbitration queue.

5.4.1.1.6 ArbQueueEntryIndicator (Class)

The ArbQueueEntryIndicator contains data necessary to specify a unique ArbQueueEntry object; in addition, it contains a reference to the TrafficEvent which is responsible for the entry.

5.4.1.1.7 ArbQueueEntryKey (Class)

This class contains the Traffic Event ID and RPI ID and is used to identify a specific ArbQueueEntry. In some cases (e.g., for HARNotifierArbQueueEntry objects), the RPI ID is the string representing a null Identifier.

5.4.1.1.8 ArbQueueSubEntryDesc (Class)

This structure hold ArbQueueEntry "device-level detail for one "sub-device (such as a constituent HAR within a SyncHAR). It holds the ID of the device and an indication as to whether the entry is active for this particular subdevice. An ArbQueueEntry for a conglomerate device (such as a SyncHAR) will contain a list of these structures, one for each constituent HAR the entry is destined for.

5.4.1.1.9 CommFailureCode (Class)

This class defines static values to be used to specify the type of comm failure in a CommFailureData object.

5.4.1.1.10 CommFailureData (Class)

This class holds data to be passed to the CommFailureDB class to be logged in the Comm failure log in the database.

5.4.1.1.11 CommFailureDB (Class)

This class is a utility used to log an entry in the Comm Failure log table in the database. This table is used to log details about any comm failure that occurs in the system.

5.4.1.1.12 ConnectedPortInfo (Class)

This class holds data pertaining to a port that was acquired and connected via the PortLocator.

5.4.1.1.13 MessageQueue (Class)

This class represents a message queue object. It will provide the ability to add, remove, and reprioritize traffic event entries in a prioritized list.

5.4.1.1.14 ModemPortLocator (Class)

This class provides an implementation of the PortLocator's abstract connectPort() method that can connect a ModemPort that has been acquired by the PortLocator base class. This derived class logs information in the comm failure database table

relating to connection problems that may occur.

5.4.1.1.15 NTCIPUtility (Class)

This class contains common utility methods for NTCIP device communications. A large portion of this class is methods refactored from the NTCIP DMS protocol handler implementation.

5.4.1.1.16 PortLocationData (Class)

This class contains configuration data that specifies the communication server(s) to use to communicate with a device.

m_commsData - One or more objects identifying the communications server (PortManager) to use to communicate with the device, in order of preference.

m_portType - The type of port to use to communicate with the device (ISDN modem, POTS modem, direct, etc.)

m_portWaitTimeSecs - The maximum number of seconds to wait when attempting to acquire a port from a port manager.

5.4.1.1.17 PortLocator (Class)

The PortLocator is a utility class that helps one to connect to the port used by the device. The actual implementation of the operations is done by the derived classes depending on what protocol is used for communication.

5.4.1.1.18 PortManagerListEntry (Class)

This class is used by the PortLocator to map object identifiers to object references for PortManager objects.

5.4.1.1.19 VoicePortLocator (Class)

This class provides an implementation of the PortLocator's abstract connectPort() method that can connect a VoicePort that has been acquired by the PortLocator base class. This derived class logs information in the comm failure database table relating to connection problems that may occur. Since this is a telephony port which is much simpler to connect than, say, a ModemPort, there will be considerably fewer types of errors which can occur and thus be detected and reported.

5.4.1.2 PortLocatorClasses (Class Diagram)

This class diagram shows utility classes that can be used to get a free port.

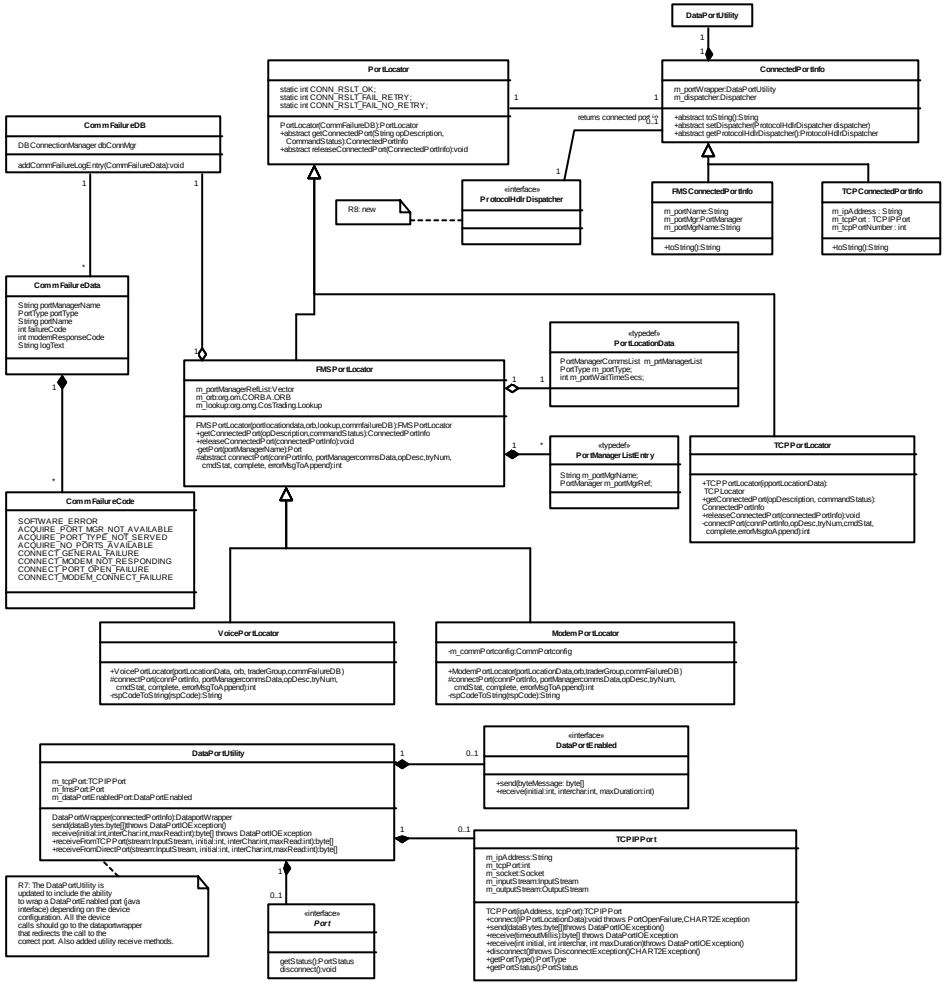


Figure 5-49. PortLocatorClasses (Class Diagram)

5.4.1.2.1 CommFailureCode (Class)

This class defines static values to be used to specify the type of comm failure in a CommFailureData object.

5.4.1.2.2 CommFailureData (Class)

This class holds data to be passed to the CommFailureDB class to be logged in the Comm failure log in the database.

5.4.1.2.3 CommFailureDB (Class)

This class is a utility used to log an entry in the Comm Failure log table in the database. This table is used to log details about any comm failure that occurs in the system.

5.4.1.2.4 ConnectedPortInfo (Class)

This class holds data pertaining to a port that was acquired and connected via the PortLocator.

5.4.1.2.5 DataPortEnabled (Class)

This interface is implemented by device specific communications classes. This interface provides an extra layer to remove dependencies on device specific packages.

5.4.1.2.6 DataPortUtility (Class)

This class is a wrapper used to hide the underlying port being used to communicate (tcp/ip ,FMS, or DataPortEnabled port).

5.4.1.2.7 FMSConnectedPortInfo (Class)

This structure defines the data used to store and exchange information about a connected port. It is returned from the PortLocator's getConnectedPort() method and is passed back into the PortLocator's release() method when it is time to release the port.

5.4.1.2.8 FMSPortLocator (Class)

The FMSPortLocator is a utility class that helps one to utilize the fault tolerance provided by the deployment of many PortManagers. The FMSPortLocator is initialized by specifying a preferred PortManager and optionally one or more alternate

PortManagers using a PortLocationData object.

When asked to get a connected port, the PortLocator first attempts to acquire a port from the preferred PortManager and then calls its abstract connectPort() method (implemented by derived classes) to attempt to connect to the port. If a failure occurs, the FMSPortLocator retries the sequence using the next PortManager in the list. The list may contain the same port manager multiple times to have retries occur on the same port manager prior to moving to another. In the event that the FMSPortLocator will perform a retry on the same port manager, it holds the previously acquired port while performing the retry to avoid having the port manager return the same port during the retry. When a different port is acquired during a retry on the same port manager, the port is released (prior to connecting the 2nd port).

5.4.1.2.9 ModemPortLocator (Class)

This class provides an implementation of the PortLocator's abstract connectPort() method that can connect a ModemPort that has been acquired by the PortLocator base class. This derived class logs information in the comm failure database table relating to connection problems that may occur.

5.4.1.2.10 Port (Class)

A Port is an object that models a physical communications resource. Derived interfaces specify various types of ports. All ports must be able to supply their status when requested.

5.4.1.2.11 PortLocationData (Class)

This class contains configuration data that specifies the communication server(s) to use to communicate with a device.

m_commsData - One or more objects identifying the communications server (PortManager) to use to communicate with the device, in order of preference.

m_portType - The type of port to use to communicate with the device (ISDN modem, POTS modem, direct, etc.)

m_portWaitTimeSecs - The maximum number of seconds to wait when attempting to acquire a port from a port manager.

5.4.1.2.12 PortLocator (Class)

The PortLocator is a utility class that helps one to connect to the port used by the device. The actual implementation of the operations is done by the derived classes depending on what protocol is used for communication.

5.4.1.2.13 PortManagerListEntry (Class)

This class is used by the PortLocator to map object identifiers to object references for PortManager objects.

5.4.1.2.14 ProtocolHdlrDispatcher (Class)

This is an empty abstract placeholder class which device specific protocol handler dispatchers must extend. Classes that extend this gain the ability to call a protocol handler method with the specific type of ConnectedPortInfo, thus reducing the usage of if/else, switch, or instance of calls, without the need to subclass into separate protocol handlers based on the ConnectedPortInfo type.

5.4.1.2.15 TCPConnectedPortInfo (Class)

This structure defines the data used to store and exchange information about a connected port. It is returned from the PortLocator's getConnectedPort() method and is passed back into the PortLocator's release() method when it is time to release the port.

5.4.1.2.16 TCPIPPort (Class)

This class provides access to a TCP/IP port for device communications.

5.4.1.2.17 TCPPortLocator (Class)

TCPPortLocator is a utility class that helps to establish and manage connection to a tcpip port.

5.4.1.2.18 VoicePortLocator (Class)

This class provides an implementation of the PortLocator's abstract connectPort() method that can connect a VoicePort that has been acquired by the PortLocator base class. This derived class logs information in the comm failure database table relating to connection problems that may occur. Since this is a telephony port which is much simpler to connect than, say, a ModemPort, there will be considerably fewer types of errors which can occur and thus be detected and reported.

5.4.2 Sequence Diagrams

5.4.2.1 AlertAndNotificationHelper:notifyAndAlert (Sequence Diagram)

This sequence diagram shows how an AlertAndNotificationHelper utility handles the task of sending alert and notification. If the new operational status equals current operational status there is nothing else to do just return. If the status has just become OK, set sendAlert flag to false (we do not send alert if status changed from COMM_FAILURE or HARDWARE_FAILURE to OK) in any other cases set flag to true to indicate that alert need to be send . Based on alert/notification settings it will decide what and whom to send.

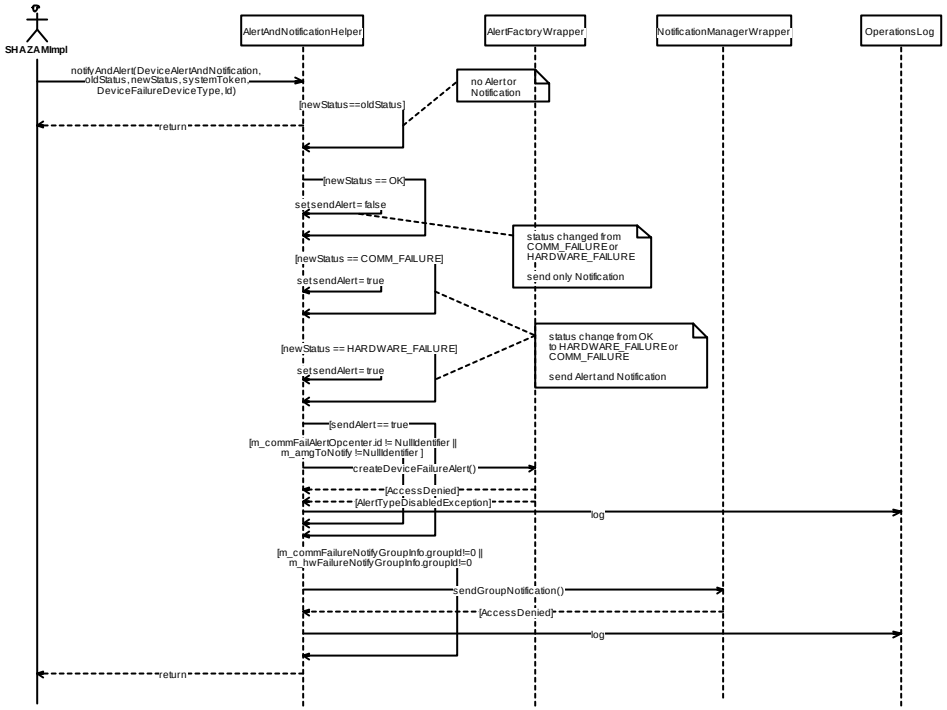


Figure 5-50. AlertAndNotificationHelper:notifyAndAlert (Sequence Diagram)

5.5 HAR Control

5.5.1 Class Diagrams

5.5.1.1 HARControlModule (Class Diagram)

This class diagram shows classes that support the use of Highway Advisory Radio (HAR) devices in the Chart II system. Details are only shown for classes that exist specifically for HAR control. Auxiliary classes used from other various utility or system interface packages are shown by name only.

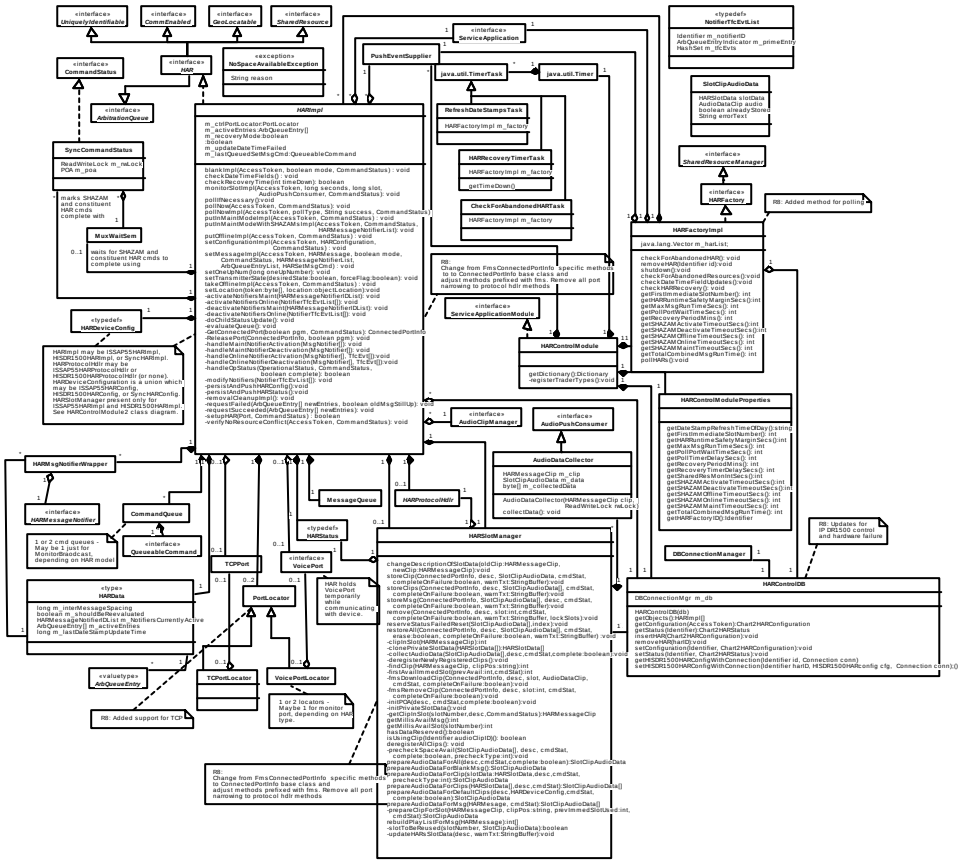


Figure 5-51 HARControlModule (Class Diagram)

5.5.1.1.1 ArbitrationQueue (Class)

An ArbitrationQueue is a queue that arbitrates the usage of a device. The evaluation of the queue determines which message(s) should be on the device, based upon the priority of the queue entries. When entries are added to the queue, they are assigned a priority level based on the type of traffic event with which they are associated, and also upon the current contents of the queue. The priority of the queue entries can be modified after they are added to the queue. The queue is evaluated when the device is online and queue entries are added or removed, when an entry's priority is modified, or when the device is put online.

5.5.1.1.2 ArbQueueEntry (Class)

This class is used for an entry on the arbitration queue, for a single message, and for a single traffic event. (It is possible, in the case of HARNotifierArbQueueEntry objects, that certain ArbQueueEntries can be on behalf of multiple TrafficEvents. In such cases, one TrafficEvent among all those involved is picked to be the responsible TrafficEvent stored in m_indicator, the ArbQueueEntryIndicator for the entry.)

5.5.1.1.3 AudioClipManager (Class)

This interface provides a way to store audio data associated with HARMessageAudioDataClip objects, converting the HARMessageAudioDataClip objects to HARMessageAudioClip objects in the process. The HARMessageAudioClip objects are created with a reference back to the AudioClipManager in them, so that the audio clips themselves can provide access to the audio data (through their stream() interface), by contacting the AudioClipManager (an AudioClipStreamer) to stream the data. The AudioClipManager also provides a capability for various AudioClipOwners to register and deregister their "interest" in a specific clip. When a clip no longer has any interested owners, it can be (and is) deleted from the database.

5.5.1.1.4 AudioDataCollector (Class)

This object is used to stream a HARMessageClip and write the streamed audio .wav data to a .wav file. It is used as a utility by the HARSlotManager to prepare HARMessageClips for download into the HAR (which is accomplished via the ISSAP55HARProtocolHdlr by passing the file name of the .wav file into it).

5.5.1.1.5 AudioPushConsumer (Class)

This interface is implemented by objects that wish to receive audio data using the push model, where the server pushes the data to the consumer. One call to pushAudioProperties() will always precede any calls to pushAudio(). When the AudioClipStreamer is done sending data in pushAudio() calls, it sends a pushCompleted() to indicate successful completion, or a pushFailure() to indicate a failure which has prevented the streaming from completing. PushAudio() returns a boolean "continue" flag, which, if returned as false, indicates that the consumer no longer wants to continue receiving audio data. In this case, the stream stops pushing data immediately, with no call to pushCompleted() or pushFailure() necessary.

5.5.1.1.6 CheckForAbandonedHARTask (Class)

This class is a timer task that is executed periodically by a timer. When the run method in this class is called, it calls the HARFactoryImpl's checkForAbandonedResources() method, which causes the factory to evaluate each HAR in the factory and issue an abandoned resource event for any HARs which have a controlling op center with no users logged in.

5.5.1.1.7 CommandQueue (Class)

The CommandQueue class provides a queue for QueueableCommand objects. The CommandQueue has a thread that it uses to process each QueueableCommand in a first in first out order. As each command object is pulled off the queue by the CommandQueue's thread, the command object's execute method is called, at which time the command performs its intended task.

5.5.1.1.8 CommandStatus (Class)

The CommandStatus CORBA interface is used to allow a calling process to be notified of the progress of a long-running asynchronous operation. This is normally used when field communications are involved to complete a method call. The most common use is to allow a GUI to show the user the progress of an operation. It can also be used and watched by a server process when it needs to call on another server process to complete an operation. The long running operation typically calls back to the CommandStatus object periodically as the command is being executed, to provide in-progress status information, and it always makes a final call to the CommandStatus when the operation has completed. The final call to the CommandStatus from the long running operation indicates the success or failure of the command.

5.5.1.1.9 CommEnabled (Class)

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of device are allowed to help in troubleshooting.

5.5.1.1.10 DBConnectionManager (Class)

This class implements a database connection manager that manages a pool of database connections. Any CHART II system thread requiring database access gets a database connection from the pool of connections maintained by this manager class. The connections are maintained in two separate lists namely, inUseList and freeList. The inUseList contains connections that have already been assigned to a thread. The freeList contains unassigned connections. This class assumes that an appropriate JDBC driver has been loaded either by using the "jdbc.drivers" system property or by loading it explicitly. The class has a monitor thread that is started by the constructor. This connection monitor thread periodically checks the inuseList to see if there are connections that are owned by dead threads and move such connections to the freeList. The connection monitor thread is started only if a non-zero value is specified for the monitoring time interval in the constructor.

5.5.1.1.11 GeoLocatable (Class)

This interface is implemented by objects that can provide location information to their users.

5.5.1.1.12 HAR (Class)

This class is used to represent a Highway Advisory Radio (HAR) device. A HAR is used to broadcast traffic related information over a localized radio transmitter, making the information available to the traveler. This interface contains methods for getting and setting configuration, getting status, changing communications modes of a HAR, and manipulating and monitoring the HAR in maintenance and online modes.

5.5.1.1.13 HARControlDB (Class)

This class contains all the database interaction for the HARControlModule. This class provides the ability to retrieve all HAR information on initialization, update of the configuration and status information, and insert or remove a HAR device from the system.

5.5.1.1.14 HARControlModule (Class)

This class implements the ServiceApplicationModule interface, providing a platform for publishing HAR objects and the HARFactory object within a service application. This class is the controlling class for the HAR module, providing for the initialization and overall operation of the module. This class creates and starts the timer tasks necessary for refreshing datestamps on the HAR, checking for abandoned shared resources, and recovery processing.

5.5.1.1.15 HARControlModuleProperties (Class)

This class contains settings from a properties file used to specify parameters to be used by objects within the HARControlModule for the current instance of the application. These settings are read during the module initialization. The module must be restarted to apply any changes made to the properties file.

5.5.1.1.16 HARData (Class)

This class is used to store and persist data pertaining to a HAR which is not part of the HARStatus (i.e., not transmitted to clients in status updates or at any other time).

5.5.1.1.17 HARDeviceConfig (Class)

HARDeviceConfig is a union which can contain the configuration for a ISS AP55 HAR, a HIS DR1500 HAR, or a Synchronizable HAR (a "virtual" HAR representing a collection of synchronized HARs). In R2B3 only DR1500 HARs are synchronizable.

5.5.1.1.18 HARFactory (Class)

This CORBA interface allows new HAR objects to be added to the system. It also allows a requester to acquire a list of HAR objects under the domain of the specific HARFactory object.

5.5.1.1.19 HARFactoryImpl (Class)

This class implements the HARFactory interface as defined by the IDL specified in the System Interfaces section. This class maintains the HAR objects served by this HAR service.

5.5.1.1.20 HARImpl (Class)

This class implements HAR as defined by IDL specified in the System Interfaces section. Since there is only one model of HAR currently envisioned for CHART II, this HARImpl class is implementing the ISS AP55 HAR specifically.

5.5.1.1.21 HARMessageNotifier (Class)

The HARMessageNotifier class specifies an interface to be implemented by devices that can be used to notify the traveler to tune in to a radio station to hear a traffic message being broadcast by a HAR. A HARMessageNotifier is directional and allows users of the device to better determine if activation of the device is warranted for the message being broadcast by the HAR. This interface can be implemented by SHAZAM devices and by DMS devices which are allowed to provide a SHAZAM-like message.

5.5.1.1.22 HARMsgNotifierWrapper (Class)

This wrapper class is used to wrap HAR message notifiers associated with a HAR. This class handles finding the reference of the notifier object given only the object's ID. The object discovery is done at the point of first use or if a currently held reference produces a CORBA failure when used.

5.5.1.1.23 HARProtocolHdlr (Class)

The HARProtocolHdlr is an abstract base class declaring methods used in communicating with a HAR device.

5.5.1.1.24 HARRecoveryTimerTask (Class)

This Timer Task runs on a regular basis (on the order of every 15-30 seconds) during the life of the process. During normal operations, this task's sole purpose is to write a timestamp to a file each time it is called. This timestamp file serves to provide, to an approximation as accurate as its frequency of invocation, when the HARService last went down, an essential piece of information for recovery during HARService startup. When the HARService has recently started up, this Task, in addition to maintaining an up-to-date timestamp in the timestamp file, also calls a method in the Factory (checkHARRecovery) which requests all HAR objects to check and see if their recovery period has expired. (The recovery period is a system-wide constant, on the order of 10-15 minutes.) Each HAR terminates its recovery period as soon as all its TrafficEvents are resolved, or when the message queue is modified through an addEntry or changePriority call, or, if neither of those cases happens, at the end of the recovery period timer. (When all HARs have terminated their recovery period, checkHARRecovery is no longer called.)

When each HAR checks its own recovery time, if it finds that it has just now exceeded the recovery period, it calls its MessageQueue to take one last try at resolving traffic events on its queue, then the HAR makes final a determination as to what message (or blank) belongs on the sign, and it requests the HAR to set its message appropriately (either to the message(s) at the top of the queue, or to the default message, if no messages are queued.

5.5.1.1.25 HARSlotManager (Class)

This class manages the slot usage for the HARImpl. When a clip is to be stored in the HAR controller, this class is called instead of calling the ISSAP55HARProtocolHdlr directly. This class ensures the reserved slot numbers (default header, default trailer, default message, immediate message slots) are not overlaid with other clips stored in the controller. When clips are stored in slots in the controller, this class keeps track of the run time for each and the total run time for the device and provides an error when the storage of a clip exceeds the configured available run time of the device.

This class also manages the condition when multiple slots are needed for the current (immediate) message. This will be true any time multiple messages are combined into one message on the HAR (up to the maximum play time for a combined message). A HAR has many immediate slots available for cases such as this.

5.5.1.1.26 HARStatus (Class)

This class (struct) contains data that indicates the current status of a HAR device. The data contained in this class is that status information which can be transmitted from the HAR to the client as necessary. This struct is also used to within the HAR Service to transmit data to/from the HARControlDB database interface class. (The HAR implementation also contains other private status data elements which are not elements of this class.)

5.5.1.1.27 java.util.Timer (Class)

This class provides asynchronous execution of tasks that are scheduled for one-time or recurring execution.

5.5.1.1.28 java.util.TimerTask (Class)

This class is an abstract base class which can be scheduled with a timer to be executed one or more times.

5.5.1.1.29 MessageQueue (Class)

This class represents a message queue object. It will provide the ability to add, remove, and reprioritize traffic event entries in a prioritized list.

5.5.1.1.30 MuxWaitSem (Class)

This object is used block execution of a thread while it is running multiple long running commands which need to be waited on. This class watches the SyncCommandStatus of each command and releases control back to the main thread when all "child" long-running processes have completed their respective CommandStatus object.

5.5.1.1.31 NoSpaceAvailableException (Class)

This exception is thrown by the HARSlotManager when there is not enough room in the HAR to store the desired message as requested. This exception is local to the HAR service only. If the exception needs to propagate out to a user (GUI), it is converted to a CHART2Exception first. The distinction is required within the HAR service since a NoSpaceAvailableException is not to be considered a failure of the device or the communications.

5.5.1.1.32 NotifierTfcEvtList (Class)

This class is used to keep track of the relationships between HAR notifiers, and the traffic events which are requesting that they be activated. One traffic event is chosen to be the primary one, and is used as part of the ArbQueueEntryIndicator stored

within this class. The m_primeEntry and m_tfcEvents are used as parameters to activate and/or modify the HAR notice on the notifier.

5.5.1.1.33 PortLocator (Class)

The PortLocator is a utility class that helps one to connect to the port used by the device. The actual implementation of the operations is done by the derived classes depending on what protocol is used for communication.

5.5.1.1.34 PushEventSupplier (Class)

This class provides a utility for application modules that push events on an event channel. The user of this class can pass a reference to the event channel factory to this object. The constructor will create a channel in the factory. The push method is used to push data on the event channel. The push method is able to detect if the event channel or its associated objects have crashed. When this occurs, a flag is set, causing the push method to attempt to reconnect the next time push is called. To avoid a supplier with a heavy supply load from causing reconnect attempts to occur too frequently, a maximum reconnect interval is used. This interval specifies the quickest reconnect interval that can be used. The push method uses this interval and the current time to determine if a reconnect should be attempted, thus reconnects can be throttled independently of a supplier's push rate.

5.5.1.1.35 QueueableCommand (Class)

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

5.5.1.1.36 RefreshDateStampsTask (Class)

This class is a timer task that is executed periodically by a timer. When executed, the run method of this class calls the HARFactoryImpl's checkDateTimeFieldUpdates(), which in turn calls each HAR in the factory to have it determine if it needs to update any field messages that use datestamp fields. These messages are reconverted to voice, and the datestamp tag, in the format "<DATESTAMP>" is replaced by text words for the day of week, month, and day of month (e.g. "Wednesday, July 14"). The reconverted messages are then queued to be resent to the HAR.

5.5.1.1.37 ServiceApplication (Class)

This interface is implemented by objects that can provide the basic services needed by a ChartII service application. These services include providing access to basic CORBA objects that are needed by service applications, such as the ORB, POA, Trader, and Event Service.

5.5.1.1.38 ServiceApplicationModule (Class)

This interface is implemented by modules that serve CORBA objects. Implementing classes are notified when their host service is initialized and when it is shutdown. The implementing class can use these notifications along with the services provided by the invoking ServiceApplication to perform actions such as object creation and publication.

5.5.1.1.39 SharedResource (Class)

The SharedResource interface is implemented by any object that may have an operations center responsible for the disposition of the resource while the resource is in use.

5.5.1.1.40 SharedResourceManager (Class)

The SharedResourceManager interface is implemented by classes that manage shared resources. Implementing classes must be able to provide a list of all shared resources under their management. Implementing classes must also be able to tell others if there are any resources under its management that are controlled by a given operations center. The shared resource manager is also responsible for periodically monitoring its shared resources to detect if the operations center controlling a resource doesn't have at least one user logged into the system. When this condition is detected, the shared resource manager must push an event on the ResourceManagement event channel to notify others of this condition.

5.5.1.1.41 SlotClipAudioData (Class)

This class is used to help keep track of and pass around slot data. This class associates a clip with a particular slot and usage, and with a file name which contains its audio (wav) data. The fileName is passed to the ISSAP55ProtocolHdlr to store the wav data in the slot.

5.5.1.1.42 SyncCommandStatus (Class)

A SyncCommandStatus implements the CommandStatus interface and performs a notification when it is completed. It is used by the HAR service to track the activity of HARMessageNotifiers, which may operate asynchronously and provide status later via a CommandStatus.

5.5.1.1.43 TCPortLocator (Class)

TCPPortLocator is a utility class that helps to establish and manage connections to a tcpip port.

5.5.1.1.44 TCPPort (Class)

This class provides access to a TCP/IP port for device communications.

5.5.1.1.45 UniquelyIdentifiable (Class)

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

5.5.1.1.46 VoicePort (Class)

A voice port provides access to a port on a telephony board. It provides methods to connect it to a destination phone number and perform send and receive operations while connected that result in DTMF or voice being sent across the telephone connection to or from the device.

5.5.1.1.47 VoicePortLocator (Class)

This class provides an implementation of the PortLocator's abstract connectPort() method that can connect a VoicePort that has been acquired by the PortLocator base class. This derived class logs information in the comm failure database table relating to connection problems that may occur. Since this is a telephony port which is much simpler to connect than, say, a ModemPort, there will be considerably fewer types of errors which can occur and thus be detected and reported.

5.5.1.2 HARControlModule2 (Class Diagram)

This class diagram shows classes that support the use of Highway Advisory Radio (HAR) devices in the Chart II system. Details are only shown for classes that exist specifically for HAR control. Auxiliary classes used from other various utility or system interface packages are shown by name only.

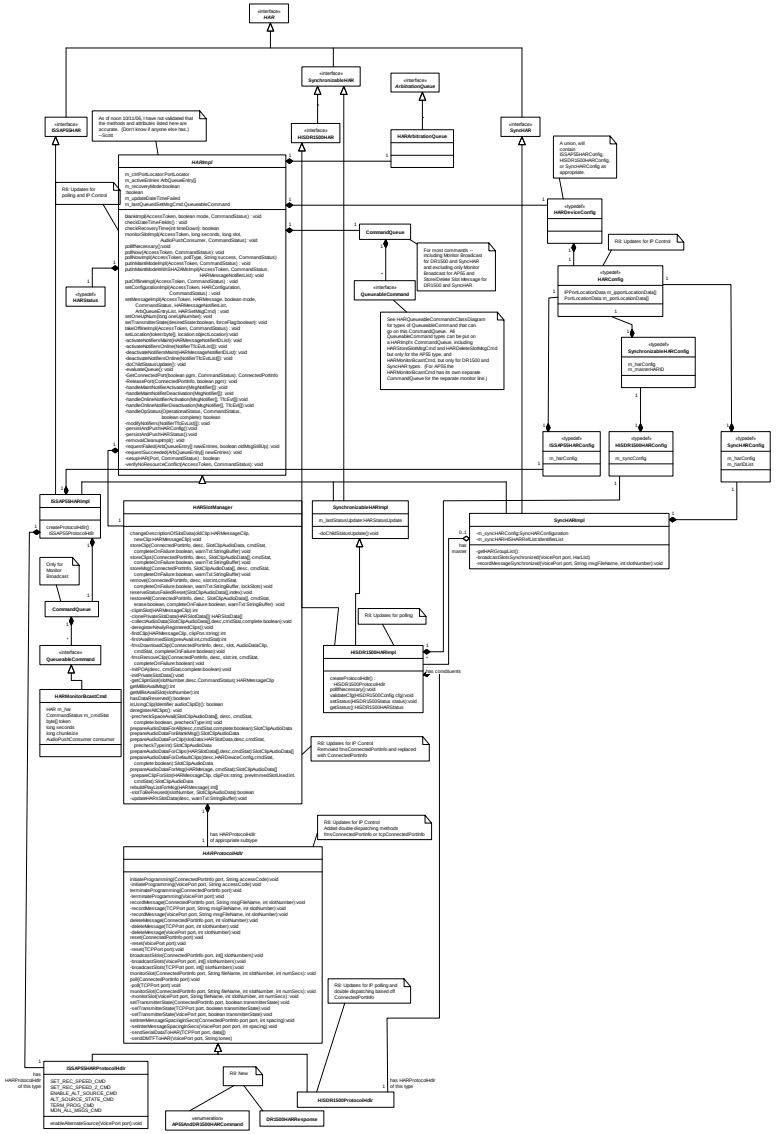


Figure 5-52. HARControlModule2 (Class Diagram)

5.5.1.2.1 AP55AndDR1500HARCommand (Class)

This enumeration class is responsible for building the DTMF string and serial byte commands for sending to AP55 and DR1500 HARs.

5.5.1.2.2 ArbitrationQueue (Class)

An ArbitrationQueue is a queue that arbitrates the usage of a device. The evaluation of the queue determines which message(s) should be on the device, based upon the priority of the queue entries. When entries are added to the queue, they are assigned a priority level based on the type of traffic event with which they are associated, and also upon the current contents of the queue. The priority of the queue entries can be modified after they are added to the queue. The queue is evaluated when the device is online and queue entries are added or removed, when an entry's priority is modified, or when the device is put online.

5.5.1.2.3 CommandQueue (Class)

The CommandQueue class provides a queue for QueueableCommand objects. The CommandQueue has a thread that it uses to process each QueueableCommand in a first in first out order. As each command object is pulled off the queue by the CommandQueue's thread, the command object's execute method is called, at which time the command performs its intended task.

5.5.1.2.4 DR1500HARResponse (Class)

This class contains helper methods for verifying and parsing DR1500 HAR responses.

5.5.1.2.5 HAR (Class)

This class is used to represent a Highway Advisory Radio (HAR) device. A HAR is used to broadcast traffic related information over a localized radio transmitter, making the information available to the traveler. This interface contains methods for getting and setting configuration, getting status, changing communications modes of a HAR, and manipulating and monitoring the HAR in maintenance and online modes.

5.5.1.2.6 HARArbitrationQueue (Class)

This class provides the implementation of an arbitration queue tailored for HAR devices.

5.5.1.2.7 HARConfig (Class)

This class holds data pertaining to a HAR device's configuration.

5.5.1.2.8 HARDeviceConfig (Class)

HARDeviceConfig is a union which can contain the configuration for a ISS AP55 HAR, a HIS DR1500 HAR, or a Synchronizable HAR (a "virtual" HAR representing a collection of synchronized HARs). In R2B3 only DR1500 HARs are synchronizable.

5.5.1.2.9 HARImpl (Class)

This class implements HAR as defined by IDL specified in the System Interfaces section. Since there is only one model of HAR currently envisioned for CHART II, this HARImpl class is implementing the ISS AP55 HAR specifically.

5.5.1.2.10 HARMonitorBcastCmd (Class)

This class contains data needed to execute a request to monitor the current message being broadcast on the HAR.

5.5.1.2.11 HARProtocolHdlr (Class)

The HARProtocolHdlr is an abstract base class declaring methods used in communicating with a HAR device.

5.5.1.2.12 HARSlotManager (Class)

This class manages the slot usage for the HARImpl. When a clip is to be stored in the HAR controller, this class is called instead of calling the ISSAP55HARProtocolHdlr directly. This class ensures the reserved slot numbers (default header, default trailer, default message, immediate message slots) are not overlaid with other clips stored in the controller. When clips are stored in slots in the controller, this class keeps track of the run time for each and the total run time for the device and provides an error when the storage of a clip exceeds the configured available run time of the device.

This class also manages the condition when multiple slots are needed for the current (immediate) message. This will be true any time multiple messages are combined into one message on the HAR (up to the maximum play time for a combined message). A HAR has many immediate slots available for cases such as this.

5.5.1.2.13 HARStatus (Class)

This class (struct) contains data that indicates the current status of a HAR device. The data contained in this class is that status information which can be transmitted from the HAR to the client as necessary. This struct is also used to within the HAR Service to transmit data to/from the HARControlDB database interface class. (The HAR implementation also contains other private status data elements which are not elements of this class.)

5.5.1.2.14 HISDR1500HAR (Class)

This interface is implemented by objects that provide for the control of an HIS model DR1500 HAR.

5.5.1.2.15 HISDR1500HARConfig (Class)

This class holds configuration data for an HIS model DR1500 HAR.

5.5.1.2.16 HISDR1500HARImpl (Class)

This class implements HISDR1500HAR as defined by IDL specified in the System Interfaces section.

5.5.1.2.17 HISDR1500ProtocolHdlr (Class)

The HISDR1500ProtocolHdlr is a class declaring methods used in communicating with a DR1500 HAR device.

5.5.1.2.18 ISSAP55HAR (Class)

This CORBA interface is implemented by objects that provide for the control of an ISS model AP55 HAR.

5.5.1.2.19 ISSAP55HARConfig (Class)

This class holds configuration data for an ISS model AP55 HAR

5.5.1.2.20 ISSAP55HARImpl (Class)

This class implements the ISSAP55HAR interfaces as defined in IDL.

5.5.1.2.21 ISSAP55HARProtocolHdlr (Class)

This protocol handler contains the protocol used to communicate with an ISS AP55 HAR device.

5.5.1.2.22 QueueableCommand (Class)

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

5.5.1.2.23 SyncHAR (Class)

This class is used to represent a synchronized Highway Advisory Radio (HAR) device. A synchronized HAR can have constituent HARs that it operates in a synchronized mode, allowing a continuous message to be delivered to the motorist as they travel out of range of one HAR and into the range of another.

5.5.1.2.24 SyncHARConfig (Class)

This class holds configuration data for a synchronized HAR.

5.5.1.2.25 SyncHARImpl (Class)

This class is implemented by objects that allow for the control of multiple HISDR1500HAR devices. Multiple HISDR1500HARs can be assigned or to one SyncHAR object we call this a SyncHAR group. The HISHARs can be active (transmitter on) or inactive (transmitter off). These devices are controlled as a group meaning the operator issues one command that is executed on all active HISHARs in the group. These comannds are HISHAR object commands. Messages can be synchronized to play on all active HISHARs simultaneously. All messages will be placed in the same slot on each active HISHAR. There is another type of command handled by the SyncHARImpl, SyncHAR object commands. SyncHAR commands create, remove or modify the SyncHARImpl object. These commands are executed on the group or SyncHAR object but not on the HISHARs which are members of the group.

5.5.1.2.26 SynchronizableHAR (Class)

This CORBA interface is implemented by objects that allow for control of HAR devices which can become constituents of a SyncHAR.

5.5.1.2.27 SynchronizableHARConfig (Class)

This class holds configuration for a HAR that can operate in a synchronized mode.

5.5.1.2.28 SynchronizableHARImpl (Class)

This class implements the SynchronizableHAR interface as defined in IDL.

5.5.1.3 HARQueueableCommandClassDiagram (Class Diagram)

This class diagram shows the classes derived from QueueableCommand necessary for HAR Control. A class exists for each type of command that can be executed asynchronously on a HAR object.

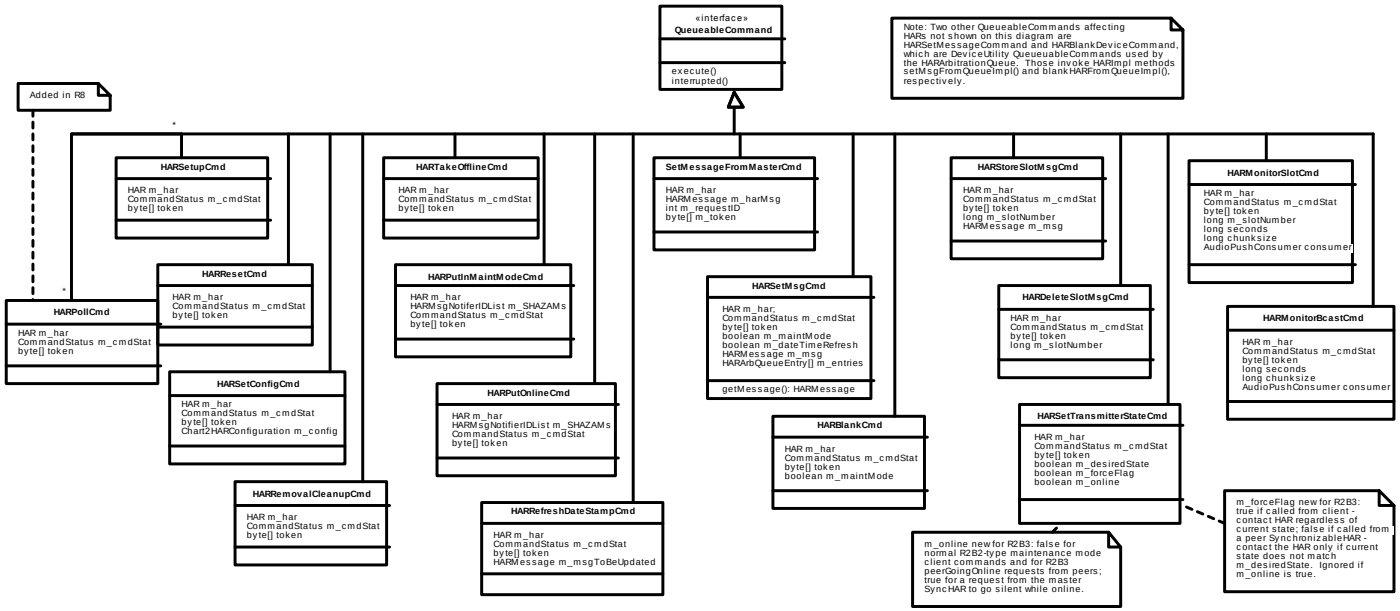


Figure 5-53. HARQueueableCommandClassDiagram (Class Diagram)

5.5.1.3.1 HARBlankCmd (Class)

This command object is used to blank the message on the HAR, which involves setting the message to the HAR's default message.

5.5.1.3.2 HARDeleteSlotMsgCmd (Class)

This class is used to hold data necessary to execute a maintenance mode request to delete a message from a slot on the HAR device.

5.5.1.3.3 HARMonitorBcastCmd (Class)

This class contains data needed to execute a request to monitor the current message being broadcast on the HAR.

5.5.1.3.4 HARMonitorSlotCmd (Class)

This class contains data needed to execute a maintenance mode request to monitor a particular slot on a HAR.

5.5.1.3.5 HARPollCmd (Class)

This class contains data needed to execute a maintenance mode request to issue the poll command for the HAR. Only IP controlled DR1500 HARS support polling.

5.5.1.3.6 HARPutInMaintModeCmd (Class)

This class contains data needed to execute a request to put a HAR into maintenance mode.

5.5.1.3.7 HARPutOnlineCmd (Class)

This class contains data needed to execute a request to put a HAR online.

5.5.1.3.8 HARRefreshDateStampCmd (Class)

This class contains data needed to execute a request to update the datestamp fields in a clip which is stored on the HAR device. This QueueableCommand, unlike most, does not reflect a user action, but reflects an action of the HARImpl itself to update its datestamp(s). The HARImpl checks for the need to update its datestamps every night shortly after midnight and queues these requests as necessary.

5.5.1.3.9 HARRemovalCleanupCmd (Class)

This class contains data needed to execute a request to clean up a HARImpl as it is being deleted. Unlike most other QueueableCommands, this command is queued by the HAR Service itself, not by a client. When a HAR is removed from the CHART II system, it may have any number of HARMessageAudioClips stored in it, and the HAR's interest in those clips needs to be deregistered with the Audio Clip Service. Rather than do this synchronously as the client request to remove the HAR is being processed, the client request is processed quickly by queuing this command for the HAR. This command, being the only command on the CommandQueue at this point, will then immediately be executed. When the command completes, this command, the HAR, and its CommandQueue will be deleted, in order, having no other references to them, and the HAR's removal will have been completed.

5.5.1.3.10 HARResetCmd (Class)

This class contains data needed to execute a maintenance mode request to reset a HAR controller.

5.5.1.3.11 HARSetConfigCmd (Class)

This class contains data needed to execute a request to change the configuration values of a HAR.

5.5.1.3.12 HARSetMsgCmd (Class)

This class contains data needed to execute a request to set the message played on a HAR. A flag is used to indicate if the message was set via a maintenance mode command or via the arbitration queue.

5.5.1.3.13 HARSetTransmitterStateCmd (Class)

This class contains data needed to execute a request to change the state (on or off) of the transmitter on a HAR device. This class also contains data needed for a SynchronizedHAR call to ensureTransmitterOff(), called when the SynchronizedHAR goes online, to ensure that the transmitter of all its peer SynchronizableHAR devices is off. The m_forcedFlag is new for R2B3. This flag is true if called from the client (the HAR is contacted regardless of the current state of the transmitter), and false if called from a peer SynchronizableHAR (the HAR is contacted only if the current state does not match the desired state (off)). Another use for this command is to queue and execute a request from a SynchronizableHAR's master SyncHAR to become silent (transmitter off) when the SyncHAR arbitration queue so dictates, based ot the transmitters (constituent HARs) selected by the user in the highest priority ArbQueueEntry.

5.5.1.3.14 HARSetupCmd (Class)

This class contains data needed to execute a maintenance mode request to issue the setup command for the HAR.

5.5.1.3.15 HARStoreSlotMsgCmd (Class)

This class contains data needed to execute a maintenance mode request to store a message clip into a slot within the HAR controller.

5.5.1.3.16 HARTakeOfflineCmd (Class)

This class contains data needed to execute a request to take a HAR offline.

5.5.1.3.17 QueueableCommand (Class)

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

5.5.1.3.18 SetMessageFromMasterCmd (Class)

This class contains data needed to execute a request from a SyncHAR to set a message to be played on one of its own constituent SynchronizableHAR objects.

5.5.2 Sequence Diagrams

5.5.2.1 HARControlModule:PollHarInBackground (Sequence Diagram)

This diagram shows the processing that occurs when an HAR is polled from the HARFactory

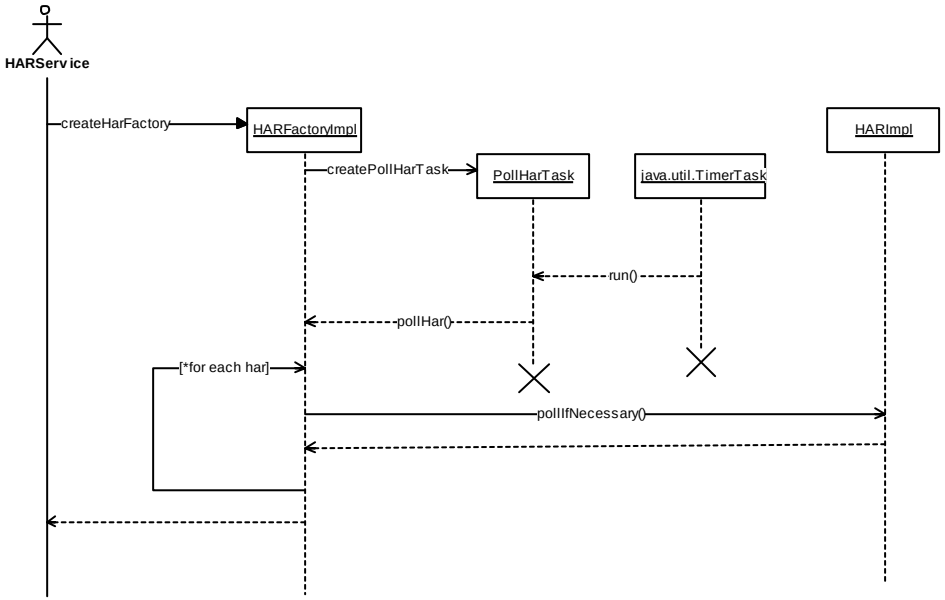


Figure 5-54. HARControlModule:PollHarInBackground (Sequence Diagram)

5.5.2.2 HARControlModule:fmsGetConnectedPort (Sequence Diagram)

This sequence diagram shows how a HARImpl object gets a voice or tcp/ip connected port. This method is called from many methods in the HAR service, whenever communications to the device is needed. A TCP/IP port is obtained from the control port locator object if it's a TCPPortLocator instance. A voice port is obtained from one of the HARImpl's two

VoicePortLocator objects if the control port locator object is a VoicePortLocator instance. For voice ports, depending on the "control" flag passed in: if control is true, the standard control (i.e., programming) port locator is used; if false, the monitor port locator is used. (The monitor port is used only for the MonitorBroadcast function for ap55 devices.) If a control port is requested, on failure a call is made to the helper method handleOpStatus to deal with the case where the operational status has changed. The CommandStatus is either updated or completed during the call to the PortLocator object based on a flag passed into this method. If the control flag is true and the device is configured to use a voice port, the HARProtocolHdlr method initiateProgramming() is called, which enters the access code ("PIN") DTMF tones, etc., in order to into the HAR and be reaady to execute programming commands on the HAR. This method is used by both the ISSAP55HAR and the HISDR1500HAR, and the HARProtocolHandler shown in the diagram will be the one appropriate for the type of HAR.

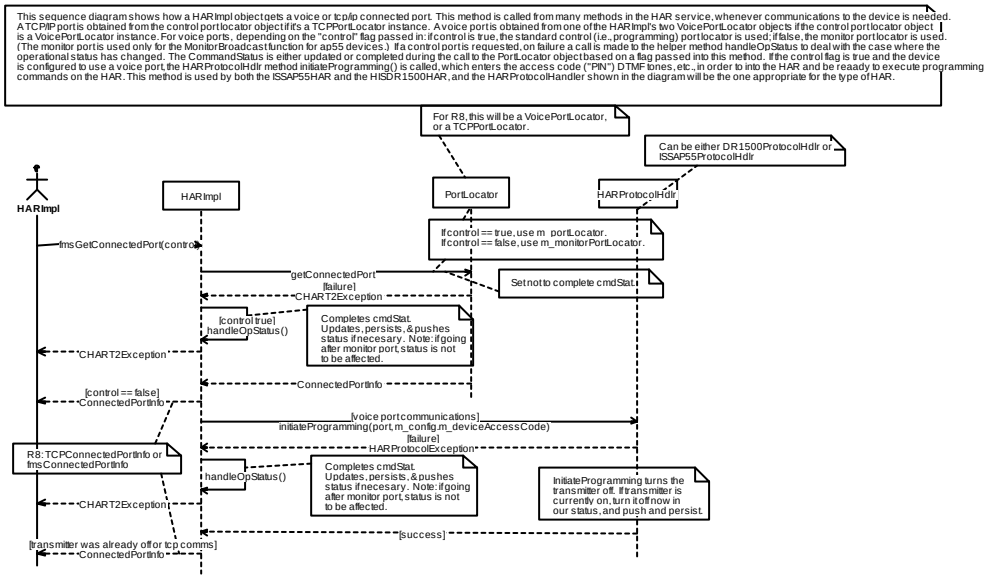


Figure 5-55. HARControlModule:fmsGetConnectedPort (Sequence Diagram)

5.5.2.3 HARControlModule:fmsReleasePort (Sequence Diagram)

This helper method releases an FMSor TCP port which is no longer needed. First, if the "control" flag is true and the port is an FMS port, the HARProtocolHdlr method terminateProgramming() is called to punch in the final DTMF tones to inform the HAR that we are done with the call and ready to hang up. It disconnects the port, and finally calls the correct PortLocator (control or monitor) or to release the port back into the pool. Errors are logged, but not reported on the CommandStatus, as the port will be released or reclaimed in any case, and errors relating to releasing a port would mask an otherwise successful status or more a useful error status.This method is used by both the ISSAP55HAR and the HISDR1500HAR, and the HARProtocolHandler shown in the diagram will be the one appropriate for the type of HAR

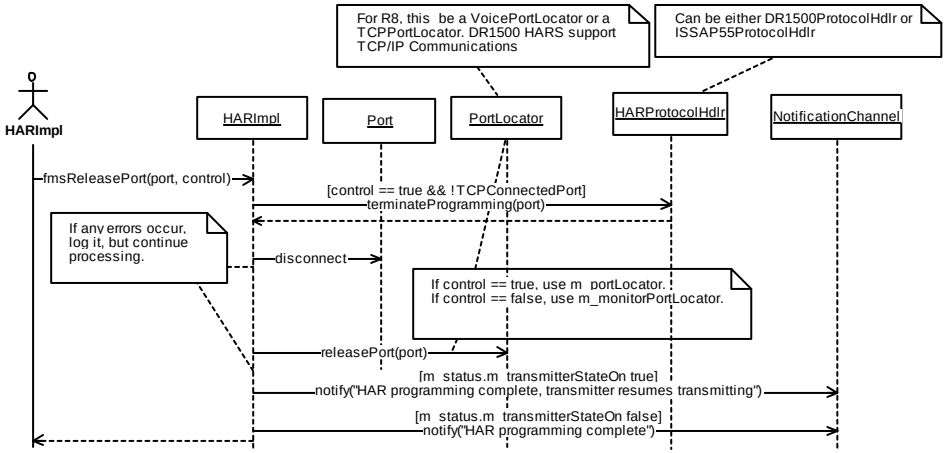


Figure 5-56. HARControlModule:fmsReleasePort (Sequence Diagram)

5.5.2.4 HARControlModule:pollHARs (Sequence Diagram)

This diagram shows the processing that occurs when hars are routinely polled by the system. For R8, only DR1500 hars communicating over TCP protocol are pollable. HARs are polled either routinely by the system, on demand from the user, or by a HAR,to confirm har status after a setup command.

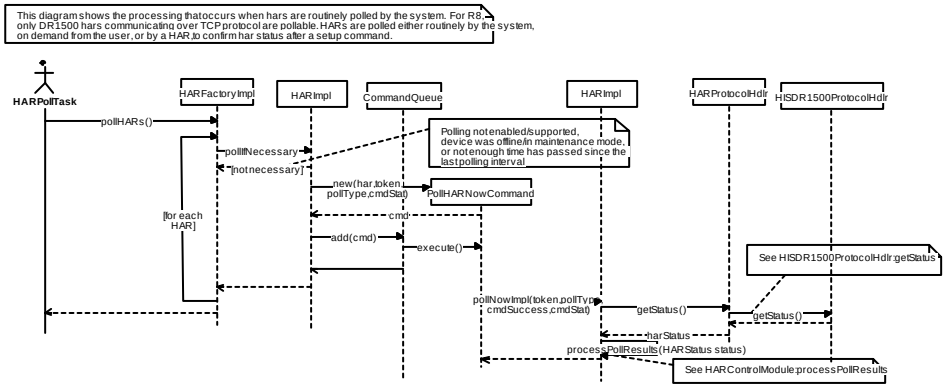


Figure 5-57. HARControlModule:pollHARs (Sequence Diagram)

5.5.2.5 HARControlModule:processPollResults (Sequence Diagram)

This diagram shows the processing that occurs when analyzing the poll results for a HAR. Only DR1500 Hars operating in TCP/IP mode support polling. If status results fall outside of configured hardware failure thresholds, the operational status will change and alerts/notifications will be sent if configured. If a playlist,transmitter, or control timestamp mismatch is detected, a setup har command will be issued.

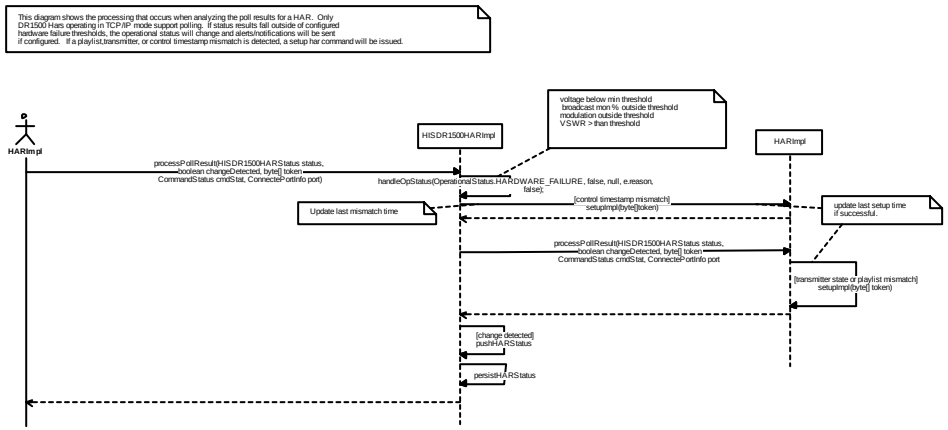


Figure 5-58. HARControlModule:processPollResults (Sequence Diagram)

5.5.2.6 HARControlModule:slotMgrStore (Sequence Diagram)

This helper method is used by the HARSlotManager to store one clip in one slot. This may be for an immediate message to be immediately played on the HAR, or it may be for the default header, message, or trailer, or it may be for a prestored slot being stored now on the HAR for possible broadcast at a later time. This method first determines whether the clip will fit, based on the run time returned by the clip itself (which could be an estimate for a text clip which has never been previewed). If the clip is deemed not to fit, a NoSpaceAvailableException is returned, otherwise, This method is used by both the ISSAP55HAR and the HISDR1500HAR, and the HARProtocolHandler shown in the diagram will be the one appropriate for the type of HAR to record the message, stored in the file specified, to the HAR. If the slot previously had a clip in it, the HAR's interest in that clip is deregistered, if it is an audio clip. The slot data for the HAR is updated and the method returns.

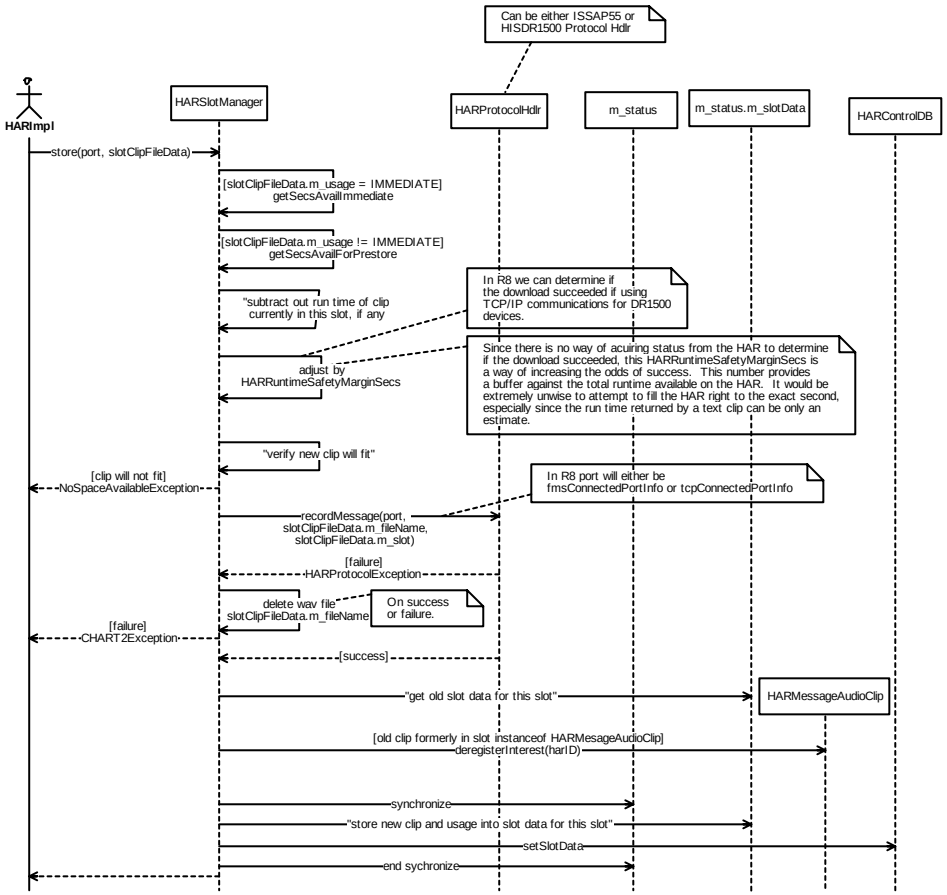


Figure 5-59. HARControlModule:slotMgrStore (Sequence Diagram)

5.5.2.7 HARControlModule:DBdeleteHAR (Sequence Diagram)

This diagram shows the processing performed to delete a HAR from the database. Data related to the HAR is deleted from the following tables: HAR_MSG_CLIP, HAR_MSG, HAR_STATUS, OBJECT_LOCATION, and DEVICE_ALERT_NOTIFICATION, and HAR. The database statements are executed as a transaction to ensure the database is not left in an inconsitent state if a statement fails.

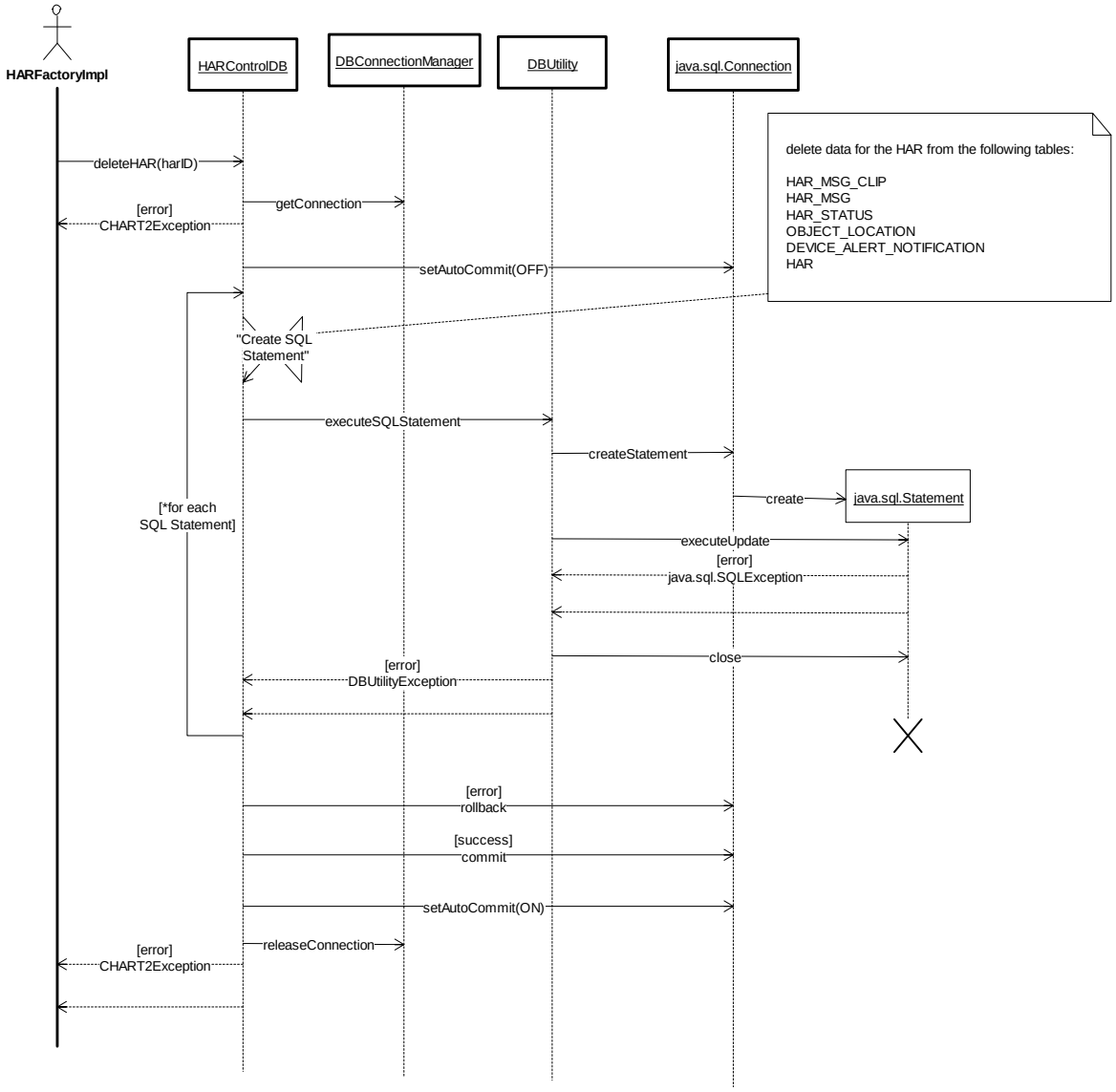


Figure 5-60 HARControlModule:DBdeleteHAR (Sequence Diagram)

5.6 HAR Protocols

5.6.1 Class Diagrams

5.6.1.1 HARProtocolsPkg (Class Diagram)

This class diagram shows the protocol handler classes that are related to HAR control.

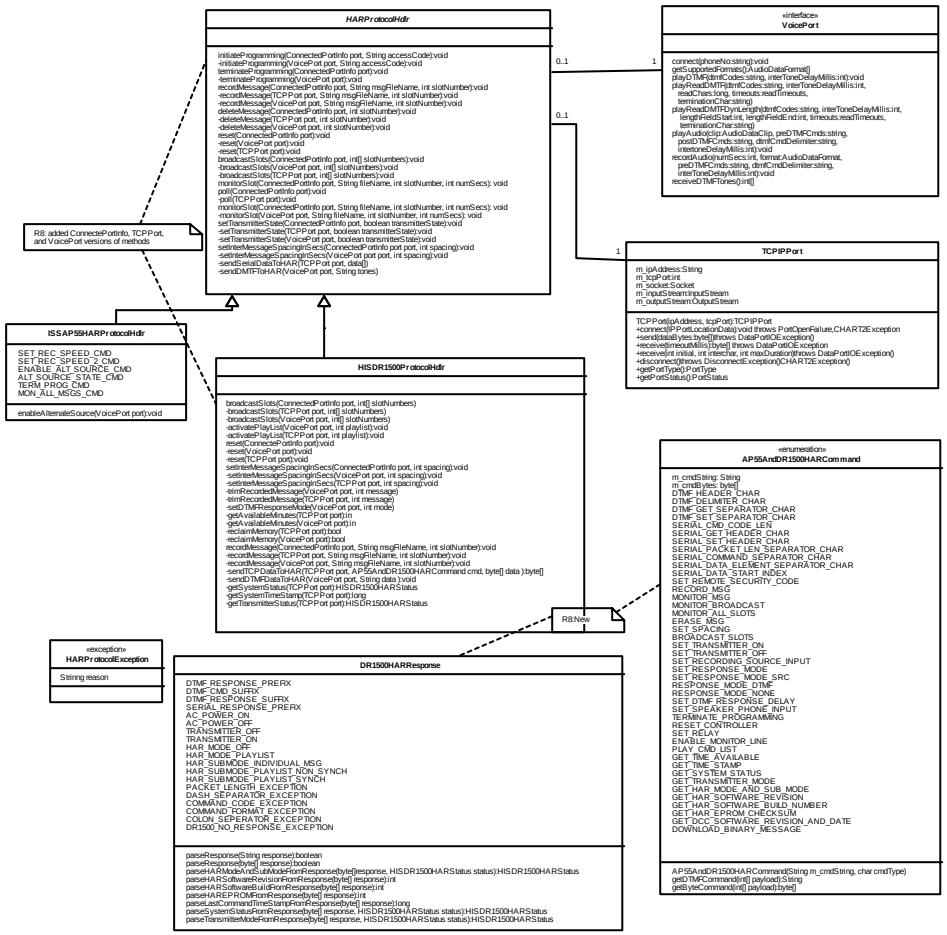


Figure 5-61 HARProtocolsPkg (Class Diagram)

5.6.1.1.1 AP55AndDR1500HARCommand (Class)

This enumeration class is responsible for building the DTMF string and serial byte commands for sending to AP55 and DR1500 HARs.

5.6.1.1.2 DR1500HARResponse (Class)

This class contains helper methods for verifying and parsing DR1500 HAR responses.

5.6.1.1.3 HARProtocolException (Class)

This class represents an exception that is thrown by HAR protocol classes when an unexpected error is encountered.

5.6.1.1.4 HARProtocolHdlr (Class)

The HARProtocolHdlr is an abstract base class declaring methods used in communicating with a HAR device.

5.6.1.1.5 HISDR1500ProtocolHdlr (Class)

The HISDR1500ProtocolHdlr is a class declaring methods used in communicating with a DR1500 HAR device.

5.6.1.1.6 ISSAP55HARProtocolHdlr (Class)

This protocol handler contains the protocol used to communicate with an ISS AP55 HAR device.

5.6.1.1.7 TCPIPPort (Class)

This class provides access to a TCP/IP port for device communications.

5.6.1.1.8 VoicePort (Class)

A voice port provides access to a port on a telephony board. It provides methods to connect it to a destination phone number and perform send and receive operations while connected that result in DTMF or voice being sent across the telephone connection to or from the device.

5.6.2 Sequence Diagrams

5.6.2.1 AP55AndDR1500HARCommand:getByteCommand (Sequence Diagram)

This diagram shows the processing that occurs when generating an array of bytes to send over a TCP port. Commands are a stream of bytes, where each byte represents the ascii code of a character.

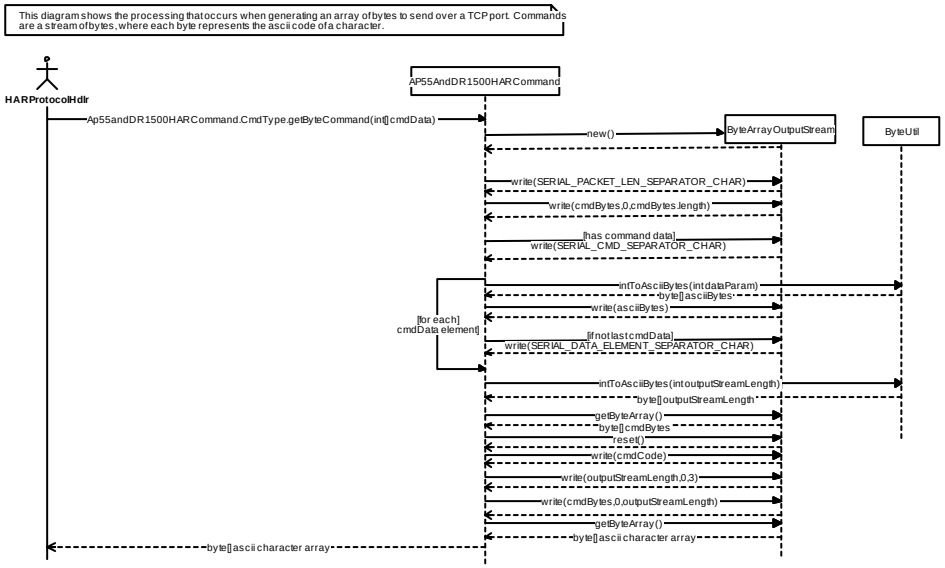


Figure 5-62 AP55AndDR1500HARCommand:getByteCommand (Sequence Diagram)

5.6.2.2 AP55AndDR1500HARCommand:getDTMFCommand (Sequence Diagram)

This diagram shows the processing that occurs when generating a string of DTMF commands to send over a voice port.

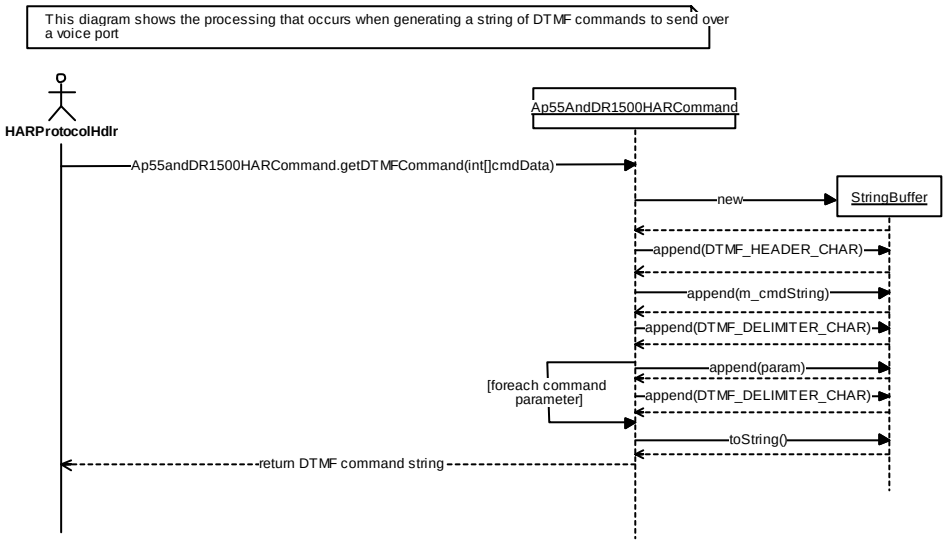


Figure 5-63. AP55AndDR1500HARCommand:getDTMFCommand (Sequence Diagram)

5.6.2.3 AP55AndDR1500HARCommand:parseLastCommandTimeStampFromResponse (Sequence Diagram)

This diagram shows the detailed processing that occurs for reading the actual ascii byte code data in the response and converting it into a long representing time since epoch.

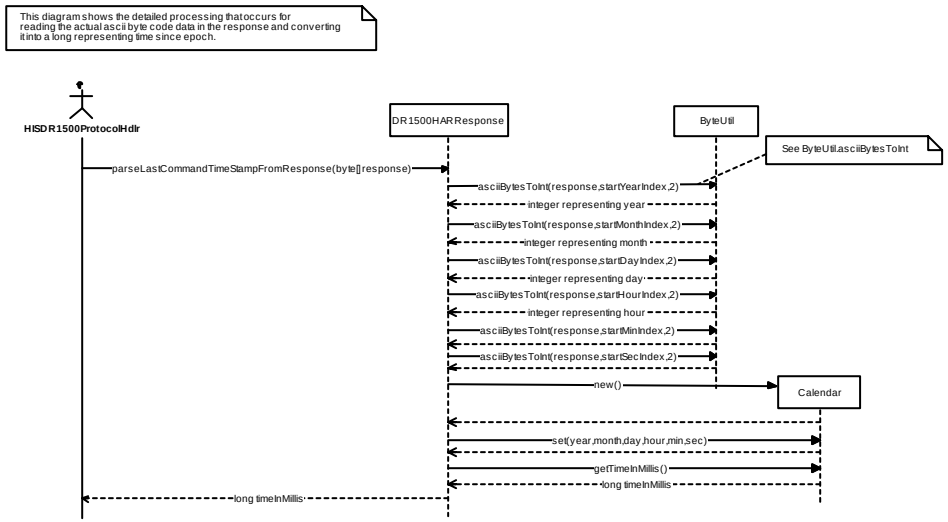


Figure 5-64. AP55AndDR1500HARCommand:parseLastCommandTimeStampFromResponse (Sequence Diagram)

5.6.2.4 HISDR1500ProtocolHdlr:BroadcastSlots (Sequence Diagram)

This sequence diagram shows the processing of broadcastSlots command of HIS DR1500 HAR protocol. For Voide port controlled devices, this involves dialing the DTMF tones for broadcast slots command (*5#) and then dialing the message number for each message to be broadcast (1#, #2, #3... for broadcasting messages 1,2,3...). For TCP/IP controlled devices, this involves building up a near DTMF equivalent byte array of ascii coded characters. A HARProtocolException is raised if an unexpected error is encountered.

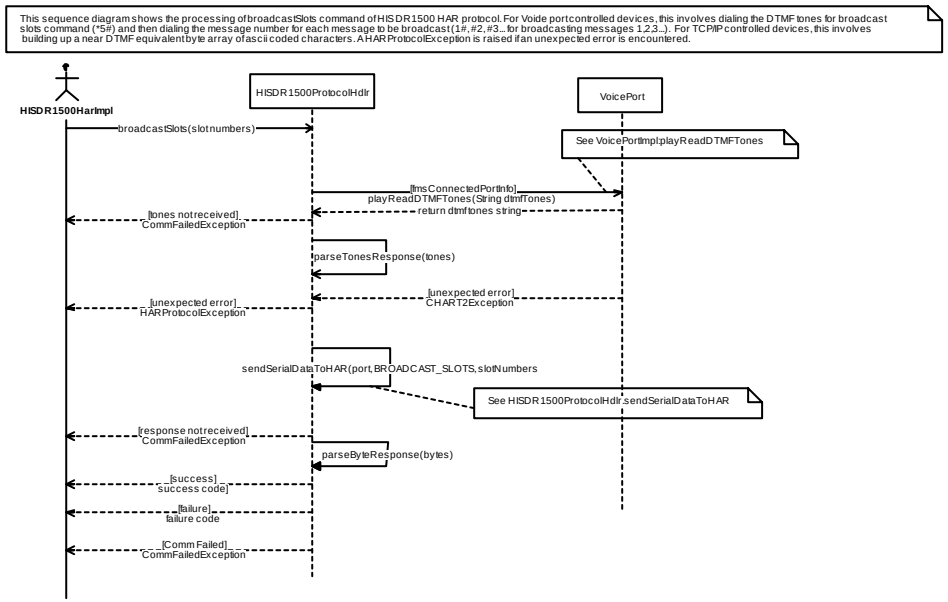


Figure 5-65. HISDR1500ProtocolHdlr:BroadcastSlots (Sequence Diagram)

5.6.2.5 HISDR1500ProtocolHdlr:getHARModeAndSubMode (Sequence Diagram)

This diagram shows the processing that occurs when the har mode, submode, and synch mode status is queried

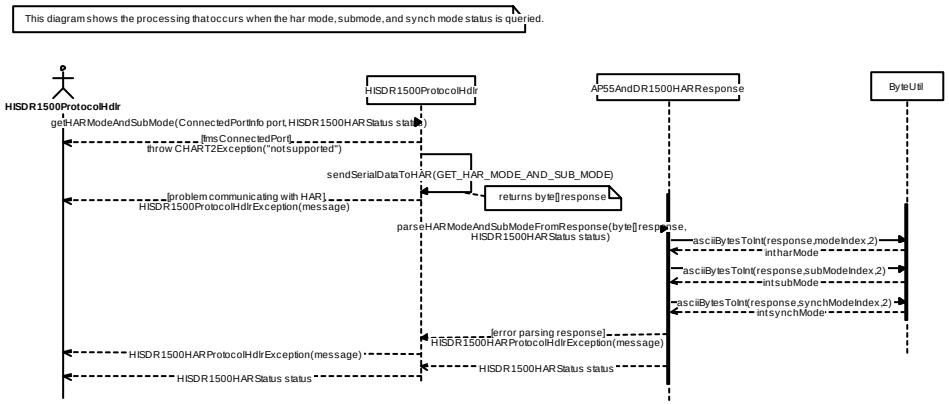


Figure 5-66. HISDR1500ProtocolHdlr:getHARModeAndSubMode (Sequence Diagram)

5.6.2.6 HISDR1500ProtocolHdlr:getLastCmdTimeStamp (Sequence Diagram)

This diagram shows the processing that occurs when a command is issued to get the last control timestamp from a DR1500 Har. This command is only supported for devices using TCP communications.

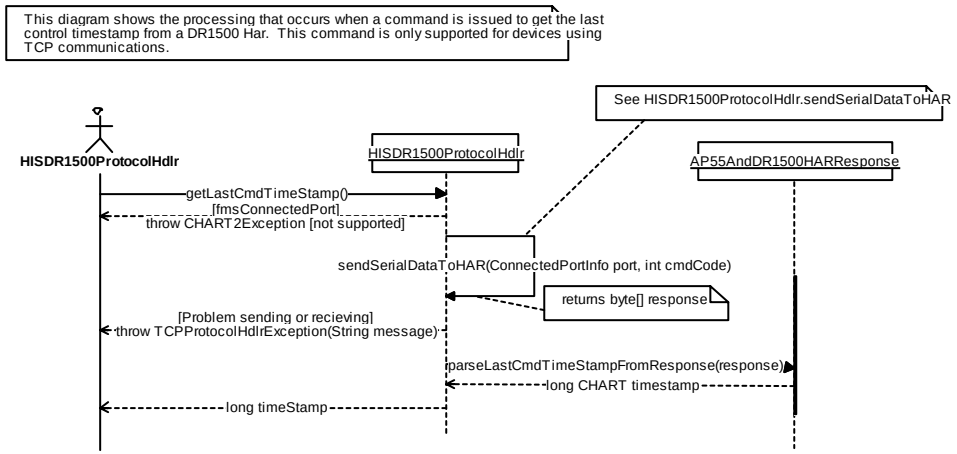


Figure 5-67. HISDR1500ProtocolHdlr:getLastCmdTimeStamp (Sequence Diagram)

5.6.2.7 HISDR1500ProtocolHdlr:getStatus (Sequence Diagram)

This diagram shows the processing that occurs when retrieving the system status from the HAR, transmitter, and DCC. We retreive various status data from the HAR to determine if the device is operating in the state CHART thinks it's in.

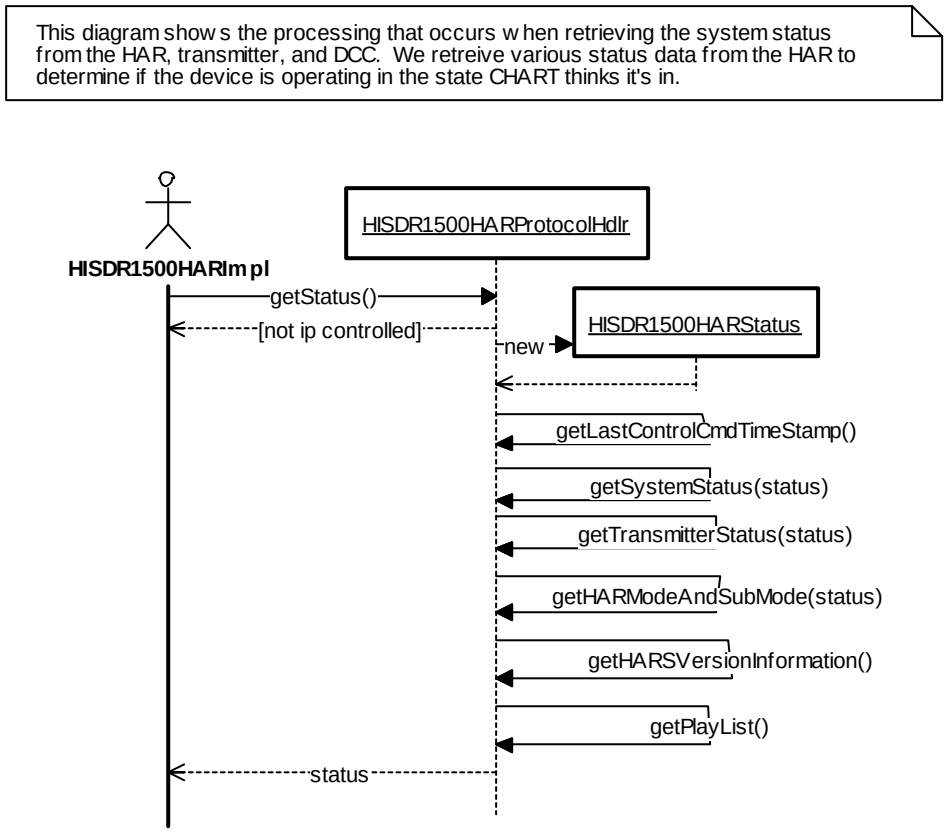


Figure 5-68. HISDR1500ProtocolHdlr:getStatus (Sequence Diagram)

5.6.2.8 HISDR1500ProtocolHdlr:getSystemStatus (Sequence Diagram)

This diagram shows the processing that occurs for retrieving the DR1500 system status. The system status includes voltage, power state, and broadcast monitor percent.

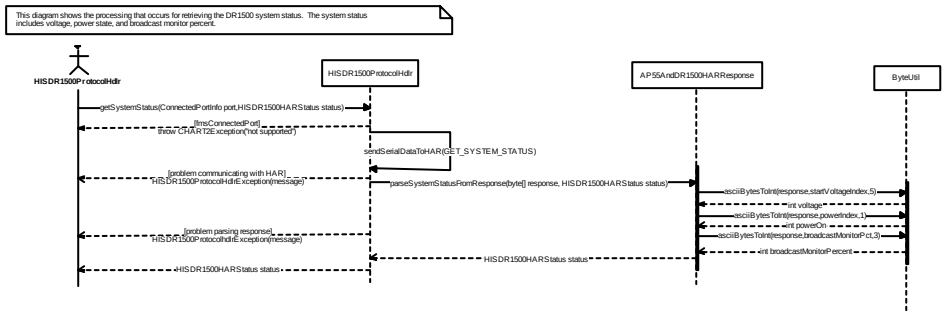


Figure 5-69. HISDR1500ProtocolHdlr:getSystemStatus (Sequence Diagram)

5.6.2.9 HISDR1500ProtocolHdlr:getTransmitterMode (Sequence Diagram)

This diagram shows the processing that occurs when the transmitter mode is queried and updated in the status. Transmitter mode is either on or off

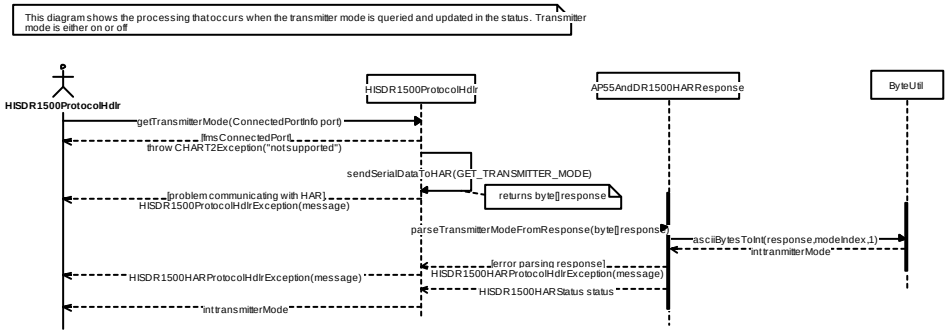


Figure 5-70. HISDR1500ProtocolHdlr:getTransmitterMode (Sequence Diagram)

5.6.2.10 HISDR1500ProtocolHdlr:getTransmitterStatus (Sequence Diagram)

This diagram shows the processing that occurs when the transmitter status is queried and updated in the status. Transmitter status includes set tx power, forward power, reflectedPower, vswr ratio, and modulation percent.

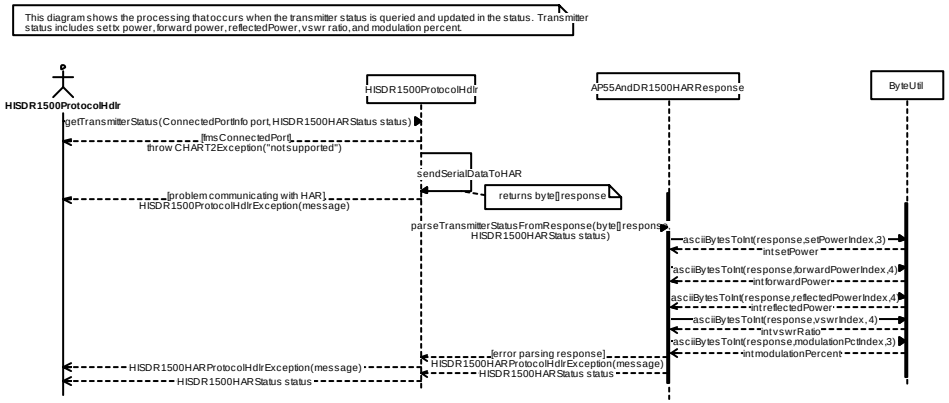


Figure 5-71. HISDR1500ProtocolHdlr:getTransmitterStatus (Sequence Diagram)

5.6.2.11 HISDR1500ProtocolHdlr:reclaimMemory (Sequence Diagram)

This diagram shows the processing that occurs when reclaiming memory on a DR1500 HAR through a DCC using TCP/IP communications.

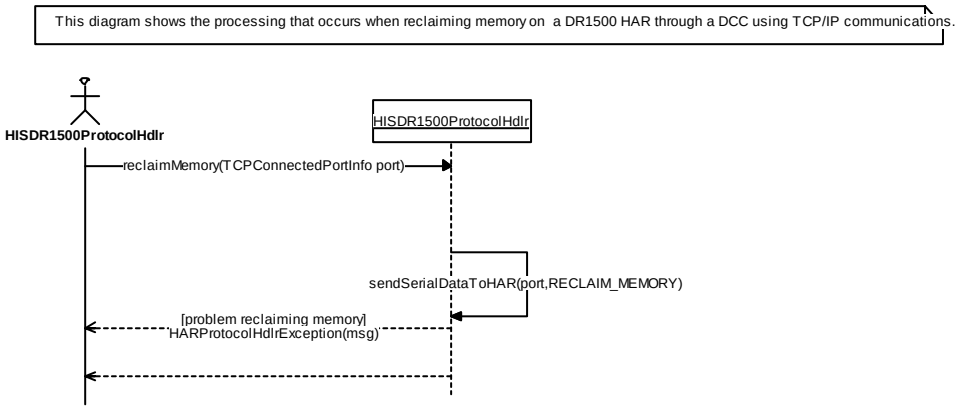


Figure 5-72. HISDR1500ProtocolHdlr:reclaimMemory (Sequence Diagram)

5.6.2.12 HISDR1500ProtocolHdlr:recordMessage (Sequence Diagram)

This diagram shows the processing that occurs when downloading a binary audio message to a DR1500 HAR through a DCC using TCP/IP communications.

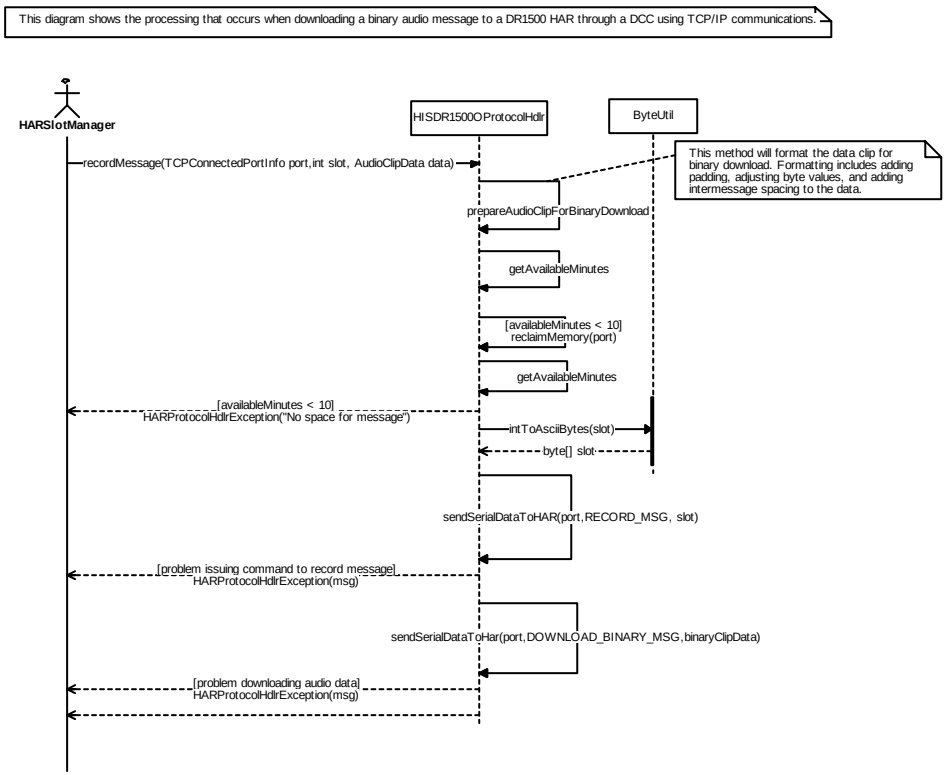


Figure 5-73. HISDR1500ProtocolHdlr:recordMessage (Sequence Diagram)

5.6.2.13 HISDR1500ProtocolHdlr:sendSerialDataToHAR (Sequence Diagram)

This diagram shows the processing that occurs when a serial command is issued to a HISDR1500HAR over a TCPPort. The command is issued. A response is received. The response is checked for validity, and returned if valid. If the response is invalid, a protocol handler exception is thrown back to the caller.

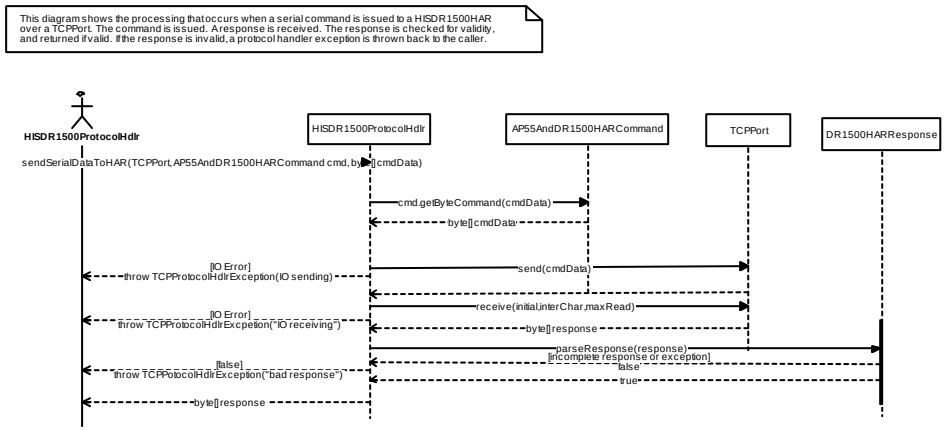


Figure 5-74. HISDR1500ProtocolHdlr:sendSerialDataToHAR (Sequence Diagram)

5.6.2.14 HISDR1500ProtocoldHdlr:parseByteResponse (Sequence Diagram)

This diagram shows the processing that occurs when parsing a byte response. The response is analyzed to determine it's a properly formatted response. After confirming the response is properly formatted, the code is compared against all known exception codes to determine if the response is successful.

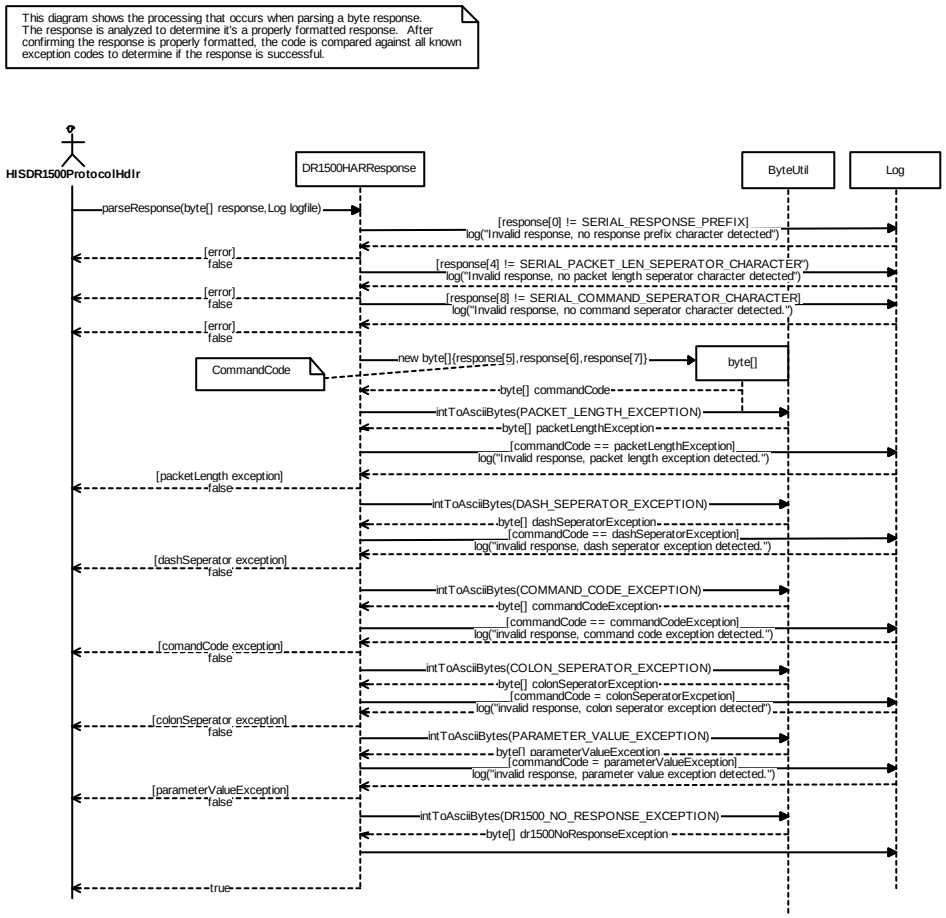


Figure 5-75. HISDR1500ProtocoldHdlr:parseByteResponse (Sequence Diagram)

5.6.2.15 HISDR1500Protocolhdlr:getHARVersionInformation (Sequence Diagram)

This diagram shows the processing that occurs when the DR1500 HAR version information is queried. DR1500 and DCC

HAR version information includes software version, build, and, eprom.

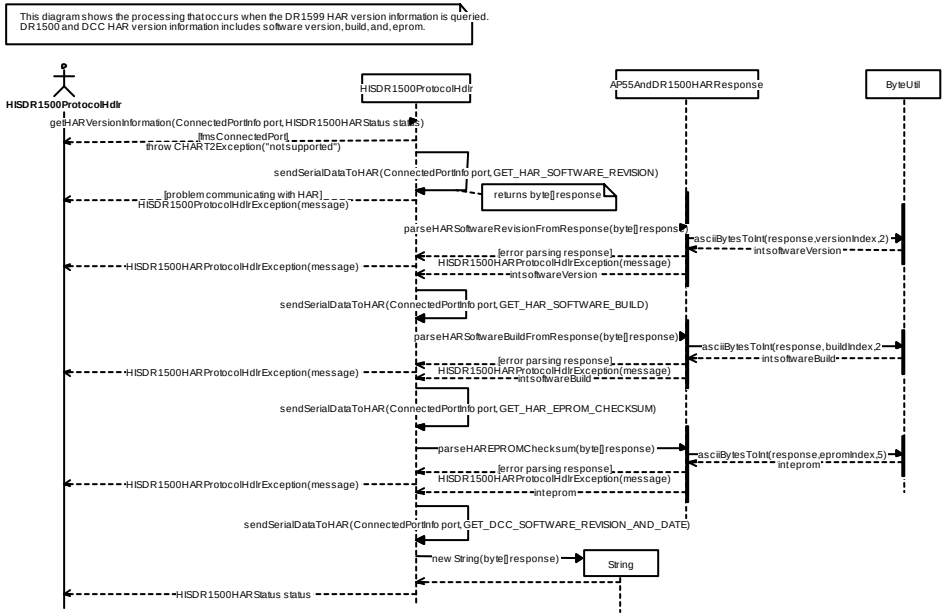


Figure 5-76. HISDR1500Protocolhdlr:getHARVersionInformation (Sequence Diagram)

5.7 SHAZAM Control Module

5.7.1 Class Diagrams

5.7.1.1 SHAZAMControl (Class Diagram)

The SHAZAMControlModule serves a SHAZAMFactory object and SHAZAM objects. The class diagram below shows the classes used to implement these system interfaces.

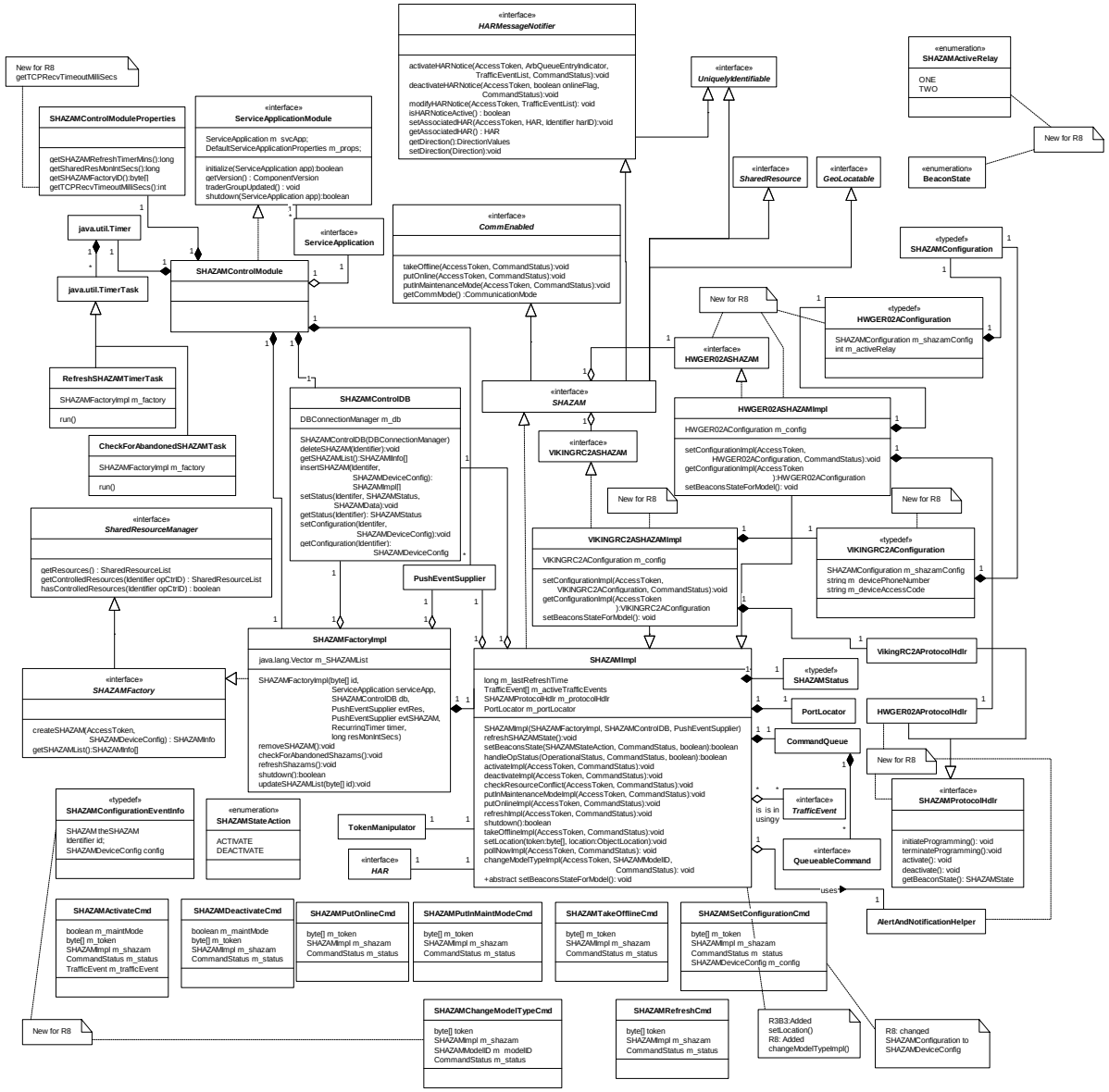


Figure 5-77. SHAZAMControl (Class Diagram)

5.7.1.1.1 AlertAndNotificationHelper (Class)

This helper class provides method to sends an alert and notification messages to a notification group.

5.7.1.1.2 BeaconState

The valid values for the current beacon state, as queried from the device.

5.7.1.1.3 CheckForAbandonedSHAZAMTask (Class)

The CheckForAbandonedSHAZAMTask class is responsible for detecting any SHAZAM device in maintenance mode with a message on it which has no one logged on at the controlling operations center. This would only occur as a result of an anomaly, such as a reboot of a user's machine, because during a normal Chart II logout attempt, the logout is prohibited by Chart II system if the the user is the last user on his/her operations center and that operations center is controlling a maintenance mode sign. However, because anomalies happen, this task runs periodically to look for abandoned SHAZAM devices. This class implements the java.util.TimerTask interface, and as such it contains one method, run(), which is invoked by Java timer object on a regularly scheduled basis. This class contains a reference to the SHAZAMFactoryImpl, which is called upon to actually check the SHAZAM objects and controlling operations centers of each SHAZAM every time this task is called.

5.7.1.1.4 CommandQueue (Class)

The CommandQueue class provides a queue for QueueableCommand objects. The CommandQueue has a thread that it uses to process each QueueableCommand in a first in first out order. As each command object is pulled off the queue by the CommandQueue's thread, the command object's execute method is called, at which time the command performs its intended task.

5.7.1.1.5 CommEnabled (Class)

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of

device are allowed to help in troubleshooting.

5.7.1.1.6 GeoLocatable (Class)

This interface is implemented by objects that can provide location information to their users.

5.7.1.1.7 HAR (Class)

This class is used to represent a Highway Advisory Radio (HAR) device. A HAR is used to broadcast traffic related information over a localized radio transmitter, making the information available to the traveler. This interface contains methods for getting and setting configuration, getting status, changing communications modes of a HAR, and manipulating and monitoring the HAR in maintenance and online modes.

5.7.1.1.8 HARMessageNotifier (Class)

The HARMessageNotifier class specifies an interface to be implemented by devices that can be used to notify the traveler to tune in to a radio station to hear a traffic message being broadcast by a HAR. A HARMessageNotifier is directional and allows users of the device to better determine if activation of the device is warranted for the message being broadcast by the HAR. This interface can be implemented by SHAZAM devices and by DMS devices which are allowed to provide a SHAZAM-like message.

5.7.1.1.9 HWGER02AConfiguration (Class)

This class contains SHAZAMConfiguration plus data that is specific to a HWG ER02a SHAZAM.

5.7.1.1.10 HWGER02AProtocolHdlr (Class)

This protocol handler contains the protocol used to communicate with a HWG ER02A SHAZAM device.

5.7.1.1.11 HWGER02ASHAZAM (Class)

This interface is used to provide access to configuration data specific to the HWG ER02a SHAZAM.

5.7.1.1.12 HWGER02ASHAZAMImpl (Class)

The HWGER02ASHAZAMImpl class provides an implementation of the HWGER02ASHAZAM interface to provide access to configuration data specific to the HWG ER02A SHAZAM.

5.7.1.1.13 java.util.Timer (Class)

This class provides asynchronous execution of tasks that are scheduled for one-time or recurring execution.

5.7.1.1.14 java.util.TimerTask (Class)

This class is an abstract base class which can be scheduled with a timer to be executed one or more times.

5.7.1.1.15 PortLocator (Class)

The PortLocator is a utility class that helps one to connect to the port used by the device. The actual implementation of the operations is done by the derived classes depending on what protocol is used for communication.

5.7.1.1.16 PushEventSupplier (Class)

This class provides a utility for application modules that push events on an event channel. The user of this class can pass a reference to the event channel factory to this object. The constructor will create a channel in the factory. The push method is used to push data on the event channel. The push method is able to detect if the event channel or its associated objects have crashed. When this occurs, a flag is set, causing the push method to attempt to reconnect the next time push is called. To avoid a supplier with a heavy supply load from causing reconnect attempts to occur too frequently, a maximum reconnect interval is used. This interval specifies the quickest reconnect interval that can be used. The push method uses this interval and the current time to determine if a reconnect should be attempted, thus reconnects can be throttled independently of a supplier's push rate.

5.7.1.1.17 QueueableCommand (Class)

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

5.7.1.1.18 RefreshSHAZAMTimerTask (Class)

The RefreshSHAZAMTimerTask class is responsible for refreshing all of the SHAZAM devices. This class implements the

java.util.TimerTask interface, and as such it contains one method, run(), which is invoked by Java timer object on a regularly scheduled basis. This class contains a reference to the SHAZAMFactoryImpl, which is called upon to request each SHAZAM to refresh itself (command the device to its last known status) if its refresh interval has expired, each time this task is called.

5.7.1.1.19 ServiceApplication (Class)

This interface is implemented by objects that can provide the basic services needed by a ChartII service application. These services include providing access to basic CORBA objects that are needed by service applications, such as the ORB, POA, Trader, and Event Service.

5.7.1.1.20 ServiceApplicationModule (Class)

This interface is implemented by modules that serve CORBA objects. Implementing classes are notified when their host service is initialized and when it is shutdown. The implementing class can use these notifications along with the services provided by the invoking ServiceApplication to perform actions such as object creation and publication.

5.7.1.1.21 SharedResource (Class)

The SharedResource interface is implemented by any object that may have an operations center responsible for the disposition of the resource while the resource is in use.

5.7.1.1.22 SharedResourceManager (Class)

The SharedResourceManager interface is implemented by classes that manage shared resources. Implementing classes must be able to provide a list of all shared resources under their management. Implementing classes must also be able to tell others if there are any resources under its management that are controlled by a given operations center. The shared resource manager is also responsible for periodically monitoring its shared resources to detect if the operations center controlling a resource doesn't have at least one user logged into the system. When this condition is detected, the shared resource manager must push an event on the ResourceManagement event channel to notify others of this condition.

5.7.1.1.23 SHAZAM (Class)

This interface class is used to identify the common SHAZAM-specific methods which can be used to interface with a SHAZAM field device. It specifies methods for activating and deactivating the SHAZAM in maintenance mode, refreshing the SHAZAM (commanding the device to its last known status) and removing the SHAZAM. This interface is implemented by a SHAZAMImpl class, which uses a helper ProtocolHdlr class to perform the model specific protocol for device command

and control.

5.7.1.1.24 SHAZAMActivateCmd (Class)

This class contains data needed to activate a SHAZAM asynchronously via the CommandQueue. A flag is used to determine if the activation is being performed directly on the device while it is in maintenance mode or if the activation is being processed as an extension of setting a HAR message in response to a traffic event.

5.7.1.1.25 SHAZAMActiveRelay (Class)

The SHAZAMActiveRelay class enumerates the types of active relay of a SHAZAM: ONE or TWO.

5.7.1.1.26 SHAZAMChangeModelTypeCmd (Class)

This class is a command object used to invoke the SHAZAM change Model type processing asynchronously from the command queue. When executed, this class calls back into the SHAZAMImpl object to execute the changeModelTypeImpl method.

5.7.1.1.27 SHAZAMConfiguration (Class)

This class contains data that specifies the configuration of a SHAZAM device. It is used to communicate configuration information to/from the database, and to/from the GUI clients. The GUI sends a SHAZAMConfiguration when creating a SHAZAM or modifying the configuration of an existing SHAZAM.Device Location member has been modified for R3B3. Now it contains a detailed location information.

5.7.1.1.28 SHAZAMConfigurationEventInfo (Class)

This class contains data (a SHAZAMDeviceConfig object) that is pushed on the SHAZAMControl CORBA event channel with a SHAZAMConfigurationChanged or SHAZAMAdded event type.

5.7.1.1.29 SHAZAMControlDB (Class)

This class provides access to database functionality needed to support the SHAZAM and SHAZAMFactory classes. This class provides a high level interface to allow for persistence and depersistance of SHAZAM and SHAZAMFactory objects.

5.7.1.1.30 SHAZAMControlModule (Class)

This class is a service module that provides control of SHAZAM devices. Upon initialization the module initializes a SHAZAMFactory which contains SHAZAM objects that have been previously added to the system. These objects are accessed via the CORBA ORB and manipulated directly from client applications. The module also creates support objects that are used by the SHAZAM (and SHAZAMFactory) objects to perform their processing, such as a database connection, event channels, and a periodic timer used to allow the objects to perform timer based processing.

5.7.1.1.31 SHAZAMControlModuleProperties (Class)

This class is used to provide access to properties used by the SHAZAM Control Module. This class wraps properties that are passed to it upon construction. It adds its own defaults and provides methods to extract properties specific to the SHAZAM Control Module.

5.7.1.1.32 SHAZAMDeactivateCmd (Class)

This class contains data needed to deactivate a SHAZAM asynchronously via the CommandQueue. A flag is used to determine if the deactivation is being performed directly on the device while it is in maintenance mode or if the deactivation is being processed as an extension of setting a HAR message in response to a traffic event.

5.7.1.1.33 SHAZAMFactory (Class)

The SHAZAMFactory class specifies the interface to be used to create SHAZAM objects within the Chart II system. It also provides a method to get a list of SHAZAM devices currently in the system.

5.7.1.1.34 SHAZAMFactoryImpl (Class)

This class provides the ability to add new SHAZAM objects to the system. When SHAZAMs are added, they are persisted to the database so this object can depersist them upon startup. This class also provides a removeSHAZAM method that allows a SHAZAM to remove itself from the system when directed. This class is also responsible for performing the checks requested by the timer tasks: to refresh the SHAZAM devices and to look for SHAZAM devices with no one logged in at the controlling operations center.

5.7.1.1.35 SHAZAMImpl (Class)

The SHAZAMImpl class provides an implementation of the SHAZAM interface, and by extension the SharedResource, HARMessageNotifier, CommEnabled, GeoLocatable, and UniquelyIdentifiable interfaces as specified by the IDL.

This class contains a CommandQueue object that is used to sequentially execute long running operations (field communications to the device) in a thread separate from the CORBA request threads, thus allowing quick initial responses.

Also contained in this class are SHAZAMConfiguration and SHAZAMStatus objects (used to store the configuration and status of the sign), a lastRefreshTime value used for refreshing (commanding the device to its last known status), and a list of TrafficEvent objects that are currently active on the SHAZAM.

The SHAZAMImpl contains *Impl methods that map to methods specified in the IDL, including requests to activate and deactivate the SHAZAM, put the SHAZAM online, put the SHAZAM offline, put the SHAZAM in maintenance mode, or to change (set) the configuration of the SHAZAM. All of these requests require (or potentially require) field communications to the device, so each request is stored in a specific subclass of QueueableCommand and added to the CommandQueue. The queueable command objects simply call the appropriate SHAZAMImpl method as the command is executed by the CommandQueue in its thread of execution.

The SHAZAMImpl also contains methods called by the SHAZAMFactory to support the timer tasks of the SHAZAM Service: to refresh the SHAZAM devices and to look for maintenance mode SHAZAM devices with no one logged in at the controlling operations center.

5.7.1.1.36 SHAZAMProtocolHdlr (Class)

This interface is used to provide methods for communicating with a SHAZAM device.

5.7.1.1.37 SHAZAMPutInMaintModeCmd (Class)

This command contains data needed to put a SHAZAM device in maintenance mode (from either offline or online mode) asynchronously via the CommandQueue. When executed this class calls back into the SHAZAMImpl object to execute the putInMaintenanceModeImpl method.

5.7.1.1.38 SHAZAMPutOnlineCmd (Class)

This command contains data needed to put a SHAZAM device online (from maintenance or offline mode) asynchronously via the CommandQueue. When executed this class calls back into the SHAZAMImpl object to execute its putOnLineImpl method.

5.7.1.1.39 SHAZAMRefreshCmd (Class)

This class is a command object used to invoke the SHAZAM refresh processing (commanding the device to its last known status) asynchronously from the command queue. When executed, this class calls back into the SHAZAMImpl object to execute the refreshImpl method.

5.7.1.1.40 SHAZAMSate (Class)

The SHAZAMSate class enumerates the types of the state of the SHAZAM ACTIVATED, DEACTIVATED and UNKNOWN.

5.7.1.1.41 SHAZAMSetConfigurationCmd (Class)

This command contains data needed to set the SHAZAM configuration asynchronously via the CommandQueue. When executed, this class calls back into the SHAZAMImpl object to execute its setConfigurationImpl method. The SHAZAM device model currently in use does not contain any configuration settings, however this command is still processed asynchronously for consistency.

5.7.1.1.42 SHAZAMStateAction (Class)

The SHAZAMStateAction class enumerates the types of actions (commands) that set the state of a SHAZAM: ACTIVATE or DEACTIVATE.

5.7.1.1.43 SHAZAMStatus (Class)

This class contains the current status of a SHAZAM device. This class is used to store status within the SHAZAM object, and is also used to communicate configuration information to/from the database, and to the GUI clients (one-way).

5.7.1.1.44 SHAZAMTakeOfflineCmd (Class)

This command contains data needed to take a SHAZAM device offline (from online or maintenance mode) asynchronously via the CommandQueue. When executed, this class calls back into the SHAZAMImpl object to execute its takeOfflineImpl method.

5.7.1.1.45 TokenManipulator (Class)

This class contains all functionality required for user rights in the system. It is the only code in the system which knows how to create, modify and check a user's functional rights. It encapsulates the contents of an octet sequence which will be passed to every secure method. Secure methods should call the checkAccess method to validate the user. Client processes should use the check access method to verify access and optimize to reduce the size of the sequence to only those rights which are necessary to invoke the secure method. The token contains the following information. Token version, Token ID, Token Time Stamp, Username, Op Center ID, Op Center IOR, functional rights

5.7.1.1.46 TrafficEvent (Class)

Objects of this type represent traffic events that require action from system operators.

5.7.1.1.47 UniquelyIdentifiable (Class)

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

5.7.1.1.48 VIKINGRC2AConfiguration (Class)

This class contains SHAZAMConfiguration plus data that is specific to a Viking RC2A SHAZAM.

5.7.1.1.49 VikingRC2AProtocolHdlr (Class)

This protocol handler contains the protocol used to communicate with a Viking RC2A SHAZAM device.

5.7.1.1.50 VIKINGRC2ASHAZAM (Class)

This interface is used to provide access to configuration data specific to the Viking RC2A SHAZAM.

5.7.1.1.51 VIKINGRC2ASHAZAMImpl (Class)

The VIKINGPC2ASHAZAMImpl class provides an implementation of the VIKINGRC2ASHAZAM interface to provide access to configuration data specific to the Viking RC2A SHAZAM.

5.7.2 Sequence Diagrams

5.7.2.1 SHAZAMControlModule:RefreshSHAZAMInBackground (Sequence Diagram)

This diagram shows the processing that is used to refresh/poll SHAZAMs periodically. When the SHAZAMControlModule is initialized it creates a timer and timer task and then schedules the timer task to fire periodically. The interval at which the timer task fires is set in the SHAZAM service's properties file, and generally needs to be set lower than the typical refresh rate of a SHAZAM, for the timer interval is the minimum rate at which SHAZAMs will be refreshed/polled. When the timer fires, the SHAZAMFactoryImpl is called and it iterates through all of its SHAZAMImpl objects and calls the refreshSHAZAMState() method in each SHAZAMImpl. The SHAZAMImpl makes several checks to determine if the SHAZAM needs to be refreshed. If the SHAZAM is not online or auto-refresh is not enabled, the SHAZAMImpl returns without doing a refresh. The last SHAZAM contact time and last SHAZAM refresh attempt time are checked to see if the SHAZAM's refresh interval has elapsed without either contacting the SHAZAM or attempting to refresh it. If the interval has not elapsed, the SHAZAMImpl returns without performing a refresh. If it is determined a refresh should be done, a SHAZAMRefreshCmd is created and added to the SHAZAM's command queue, where it will be executed asynchronously. See the refreshImpl sequence diagrams for details.

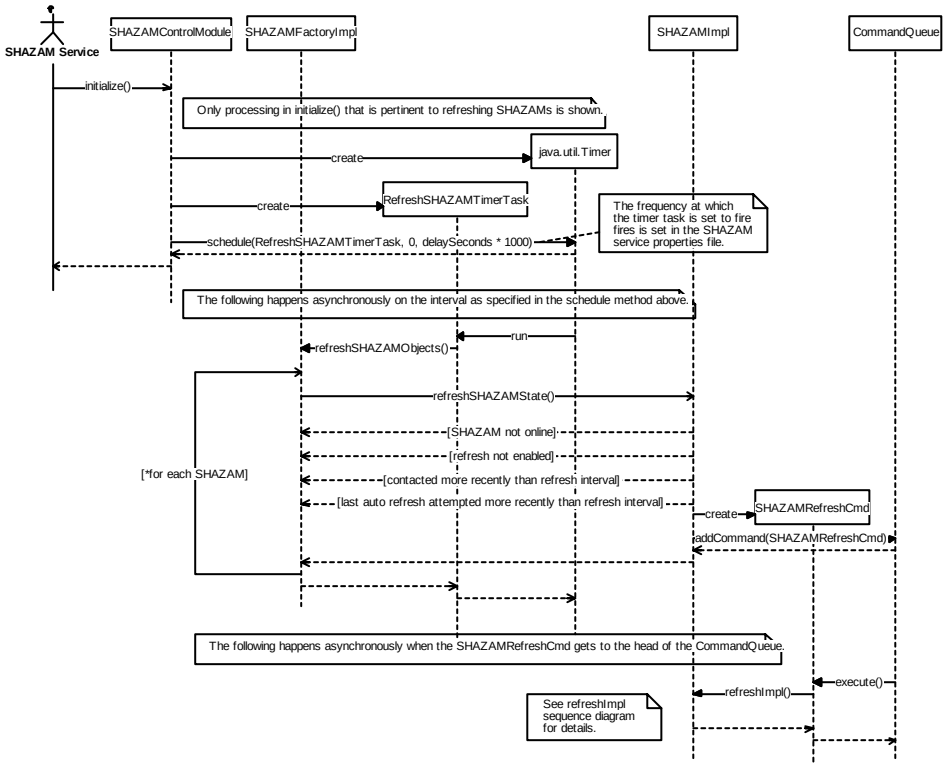


Figure 5-78. SHAZAMControlModule:RefreshSHAZAMInBackground (Sequence Diagram)

5.7.2.2 SHAZAMControlModule:changeModelType (Sequence Diagram)

A user with the proper functional rights can modifies the model type of the SHAZAM. The SHAZAM must be offline to change its model type. The SHAZAM must be offline to change its model type. If newModelID matches the current modelID, the cmdStat will be completed with success, with text indicating the model type matches, and no event will be pushed.

If the new model type is indeed different, depending on the old and new model types, this operation may have to be

implemented by deleting the old SHAZAM servant object and instantiating a new one. This will change the CORBA reference of the device, even though it will maintain the same ID. In these cases, the old SHAZAM will be deactivated and withdrawn from the Trader, and the new SHAZAM will then be published to the Trader and activated. In all cases, regardless of whether a new servant had to be created, a SHAZAMModelChanged event will be pushed, specifying the new (or the same) CORBA reference for the SHAZAM as it now exists. The current configuration of the SHAZAM matching the new SHAZAM Model type will also be included in the event. The new SHAZAM will have the same CHART ID as the SHAZAM this operation is invoked on.

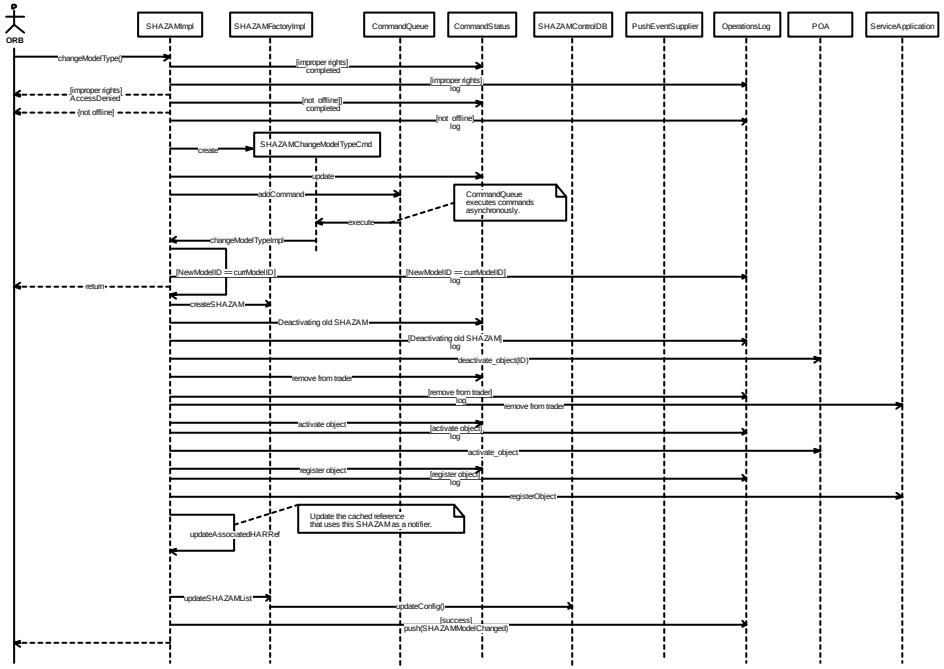


Figure 5-79. SHAZAMControlModule:changeModelType (Sequence Diagram)

5.7.2.3 SHAZAMControlModule:createSHAZAM (Sequence Diagram)

A user with the proper functional rights can add a SHAZAM to the system. The SHAZAM configuration data is added to the database, and an appropriate SHAZAMImpl object is created. Upon creation, SHAZAMImpl creates an appropriate ProtocolHdlr and a CommandQueue. The SHAZAMImpl object is connected to the POA, making it ready for calls from clients. The Service Application is called to register the object with the trader and an event is pushed on the status event channel to allow GUIs to show this SHAZAM as an available object in the system. The SHAZAM is added in the offline state and no field communications are necessary.

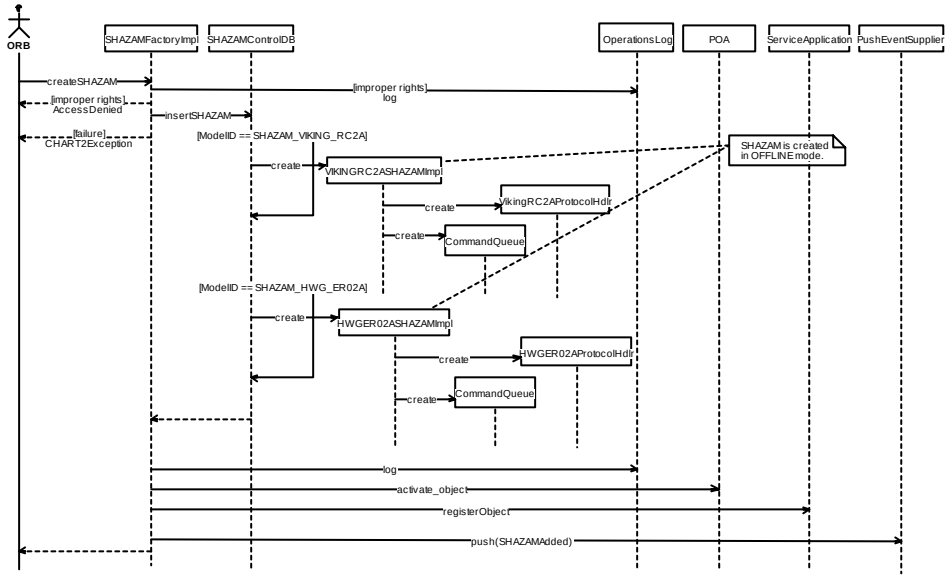


Figure 5-80. SHAZAMControlModule:createSHAZAM (Sequence Diagram)

5.7.2.4 SHAZAMControlModule:getConfiguration (Sequence Diagram)

A user with appropriate privileges can get the current configuration of the SHAZAM. When a request is made for the current

configuration of the SHAZAM, the specific (Viking or HWGER02A) SHAZAMConfiguration object is returned.

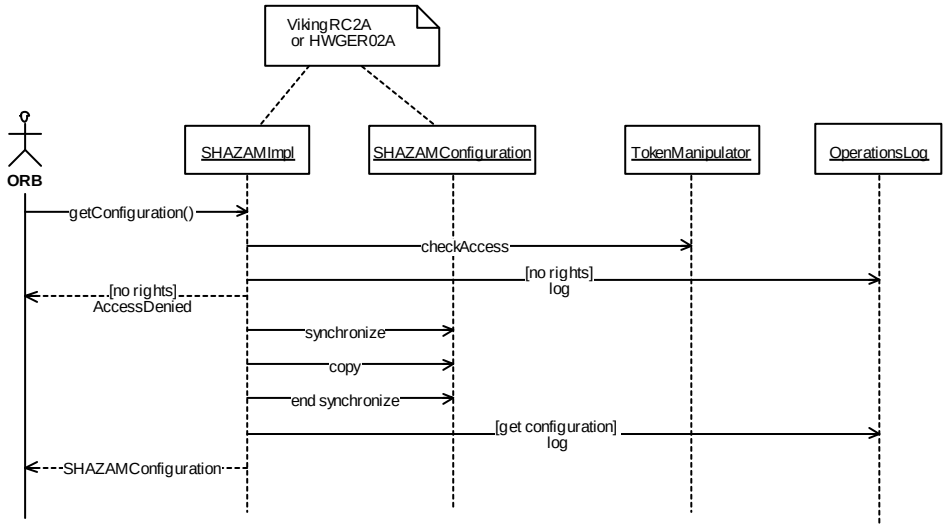


Figure 5-81. SHAZAMControlModule:getConfiguration (Sequence Diagram)

5.7.2.5 SHAZAMControlModule:getStatus (Sequence Diagram)

A user with appropriate privileges can get the current status of the SHAZAM. When a request is made for the current status of the SHAZAM, the SHAZAM's SHAZAMStatus object is returned.

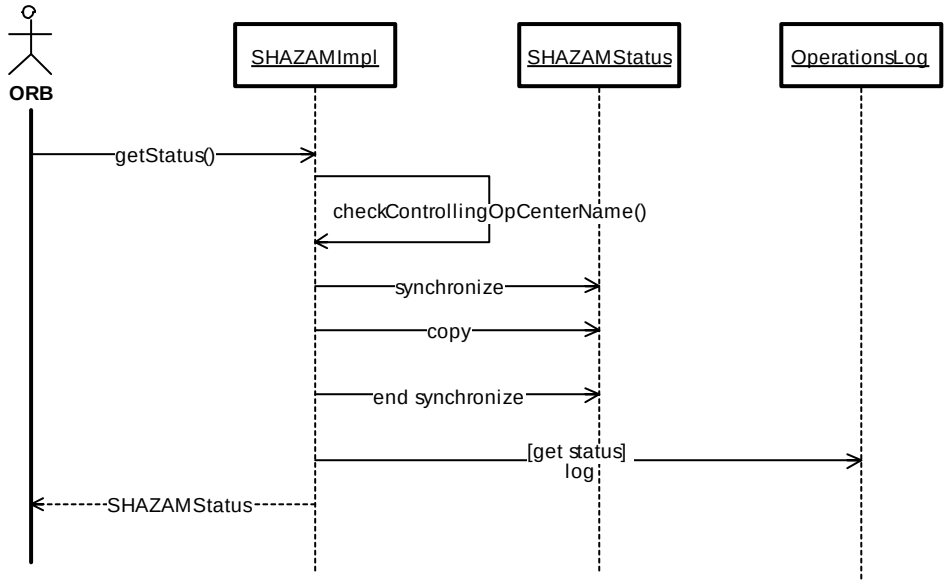


Figure 5-82. SHAZAMControlModule:getStatus (Sequence Diagram)

5.7.2.6 SHAZAMControlModule:handleOpStatus (Sequence Diagram)

This sequence diagram shows how a SHAZAMImpl handles the task of detecting and responding to changes in its operational status. A SHAZAM's operational status is normally "OK", but it can be "COMM_FAILURE" when the SHAZAMProtocolHndlr reports that it cannot communicate with the device, or HARDWARE_FAILURE when relay state does not match with a CHART state.

This method is called after every attempt to communicate with the device. Processing falls into one of two cases, depending on the operational status reported.

If the operational status is now being reported OK, the last contact time in m_status (a SHAZAMStatus object) is updated with the current time. (The last contact time is used to determine when to refresh [see runRefreshSHAZAMTask].) If the operational status of the device was already OK, there is no change in operational status and there is nothing else to do except

return false (false indicates no change in operational status). If the status has just become OK, the operational status in m_status is updated to OK, the status change time in m_status is updated to the current time, and the new SHAZAMStatus is persisted and pushed out into the status event channel. The command status is then updated or completed depending on the complete flag. This method then returns true indicating that the operational status has changed.

If the operational status is now being reported COMM_FAILURE and the device was already in COMM_FAILURE, there is no change in operational status and there is nothing else to do except return false (false indicates no change in operational status). If the status has just become COMM_FAILURE, the operational status in m_status is updated to COMM_FAILURE, the status change time in m_status is updated to the current time, and the new SHAZAMStatus is persisted and pushed out into the status event channel. The command status is then updated or completed depending on the complete flag. This method then returns true indicating that the operational status has changed.

If the operational status is now being reported HARDWARE_FAILURE and the device was already in HARDWARE_FAILURE, there is no change in operational status and there is nothing else to do except return false (false indicates no change in operational status). If the status has just become HARDWARE_FAILURE, the operational status in m_status is updated to HARDWARE_FAILURE, the status change time in m_status is updated to the current time, and the new SHAZAMStatus is persisted and pushed out into the status event channel. The command status is then updated or completed depending on the complete flag. This method then returns true indicating that the operational status has changed.

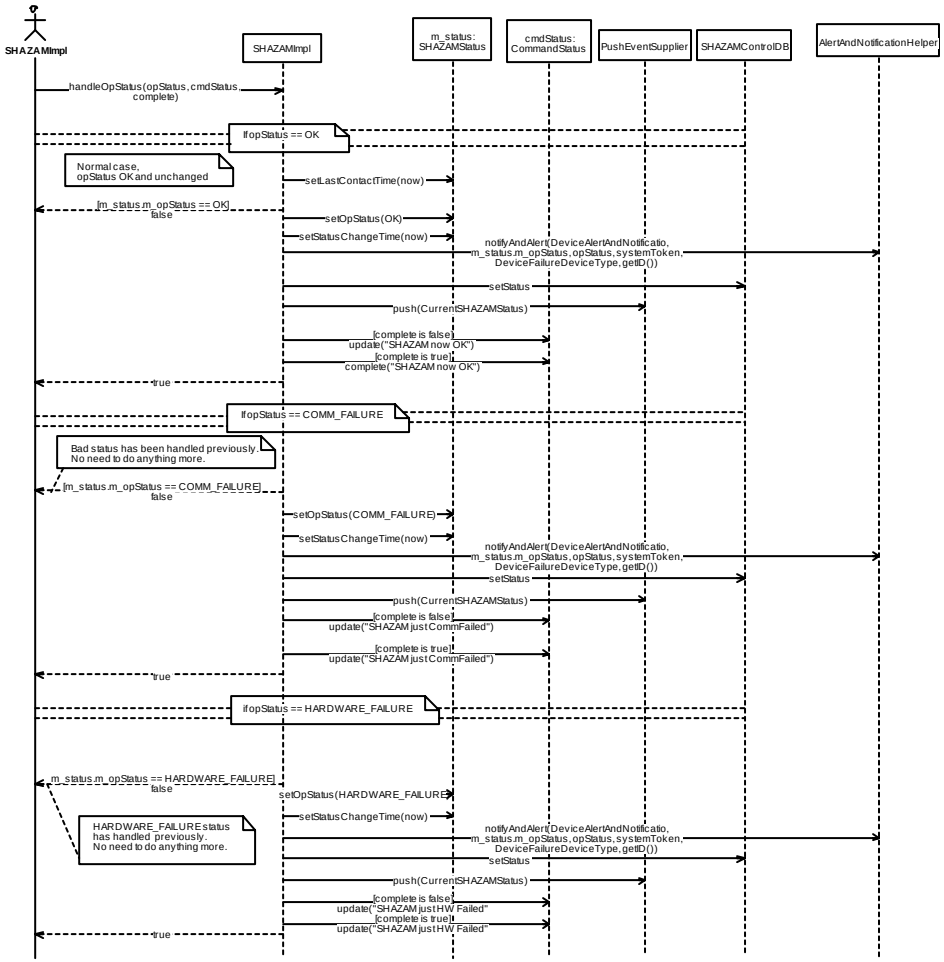


Figure 5-83. SHAZAMControlModule:handleOpStatus (Sequence Diagram)

5.7.2.7 SHAZAMControlModule:refreshImpl (Sequence Diagram)

This sequence diagram shows how a SHAZAMImpl handles the refresh task. It checks to make sure the SHAZAM is not offline and there is no resource conflict, then calls its setBeaconState() method with the desired state set to the current state as indicated in the status. See the setBeaconState sequence diagram for details.

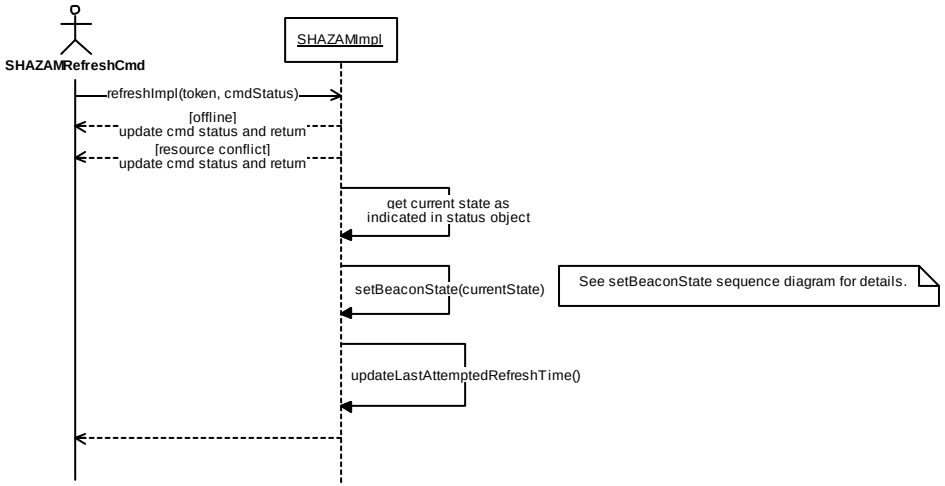


Figure 5-84. SHAZAMControlModule:refreshImpl (Sequence Diagram)

5.7.2.8 SHAZAMControlModule:setBeaconStateForModel_HWGER02A (Sequence Diagram)

This sequence diagram shows HWGER02A model specific sequence to set the beacon state. It sends the activate or deactivate SHAZAM command to the device via the protocol handler. Any exception thrown from the protocol handler (which will occur if there is no response from the device or if the response from the device is not valid) will be allowed to fly out of this method to the caller and can be treated as a communication failure.

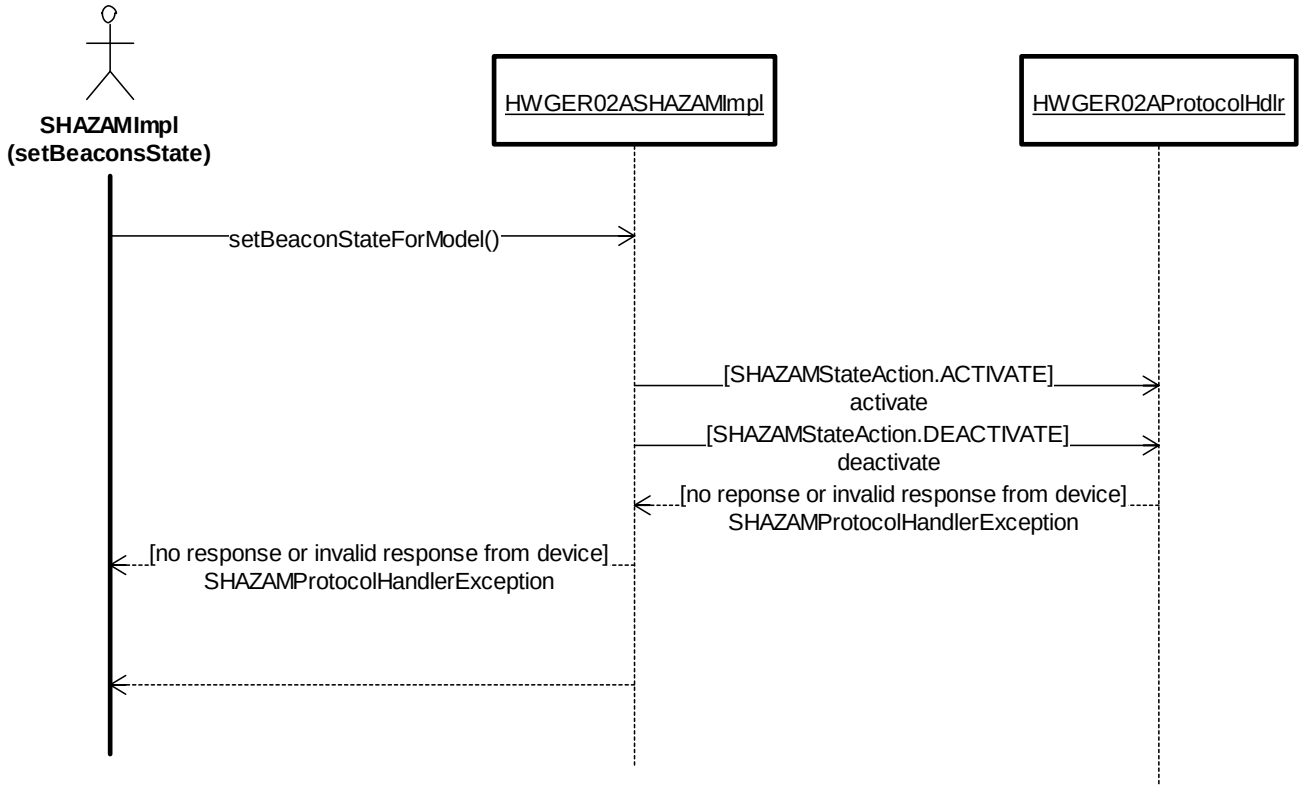


Figure 5-85. SHAZAMControlModule:setBeaconStateForModel_HWGER02A (Sequence Diagram)

5.7.2.9 SHAZAMControlModule:setBeaconStateForModel_VikingRC2A (Sequence Diagram)

This sequence diagram shows the VikingRC2A model specific sequence to activate or deactivate the SHAZAM beacons. The Viking RC2A does not provide any confirmations that commands succeeded, so the initiate and activate/deactivate commands are issued twice to increase the chance of success. If any attempt to send DTMF tones to the device fails, a protocol handler exception will be thrown by the protocol handler and this method will let that exception fly out of this method to the caller. Any such exception can be treated as a communications failure for this device.

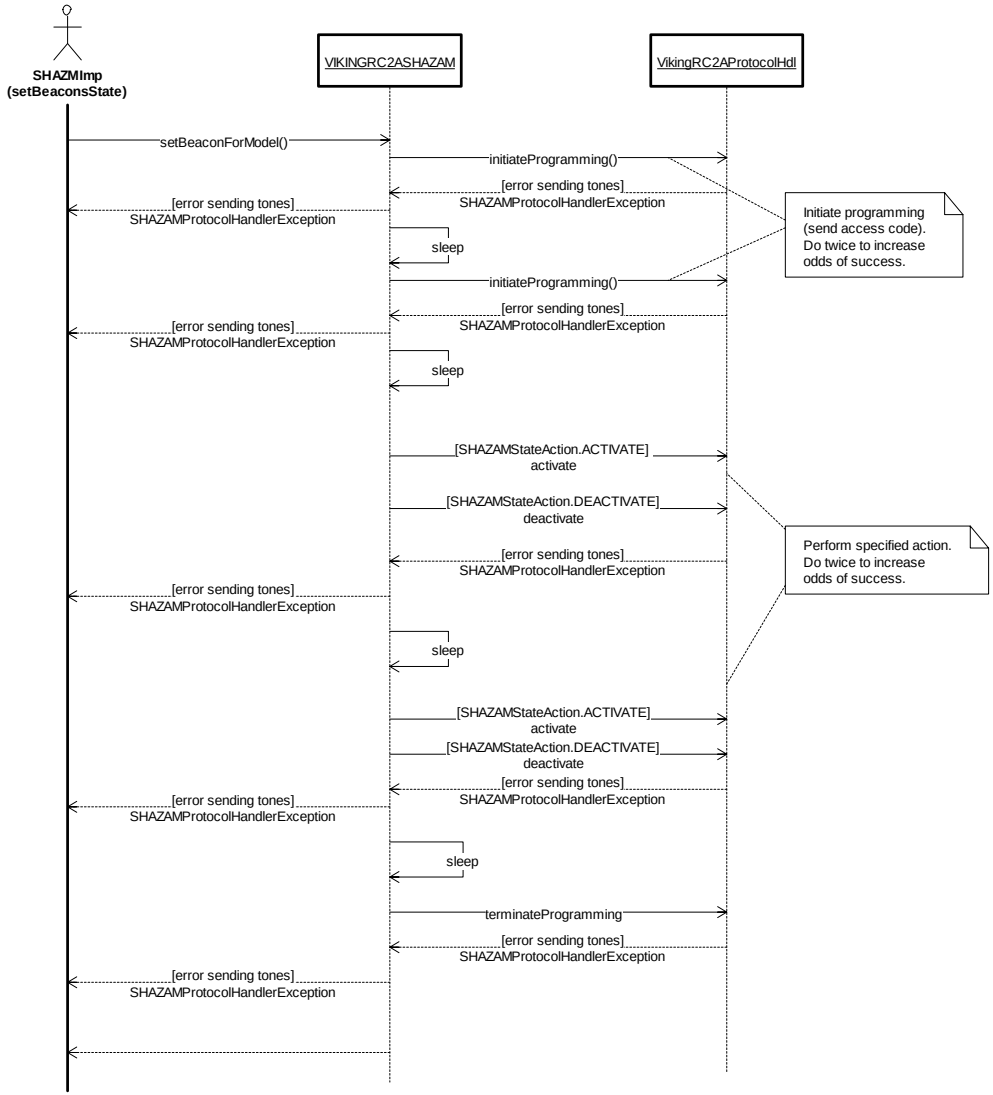


Figure 5-86. SHAZAMControlModule:setBeaconStateForModel_VikingRC2A (Sequence Diagram)

5.7.2.10 SHAZAMControlModule:setBeaconsState (Sequence Diagram)

This sequence diagram shows how a SHAZAMImpl object activates or deactivates a SHAZAM. This method is called from several methods in the SHAZAM service. A port is obtained from the PortLocator object and a call is made to the abstract setBeaconStateForModel() method which is implemented by the derived model specific class. The model specific method sends a request to either activate or deactivate via the protocol handler. If the model specific method has indicated a status of OK, a call is made to the protocol handler to retrieve the current beacon state from the device. If the model does not support this, the protocol handler will return Unknown, however if the model does support querying the status the returned state is compared against the commanded state and a mismatch will result in the SHAZAM status being set to hardware failed. A call is made to the helper method handleOpStatus to deal with the case where the operational status has changed, including initiating alerts and/or notifications as needed. The new state is stored and the SHAZAMStatus is persisted and pushed out into the status event channel. The command status is either updated or completed based on a flag passed into this method.

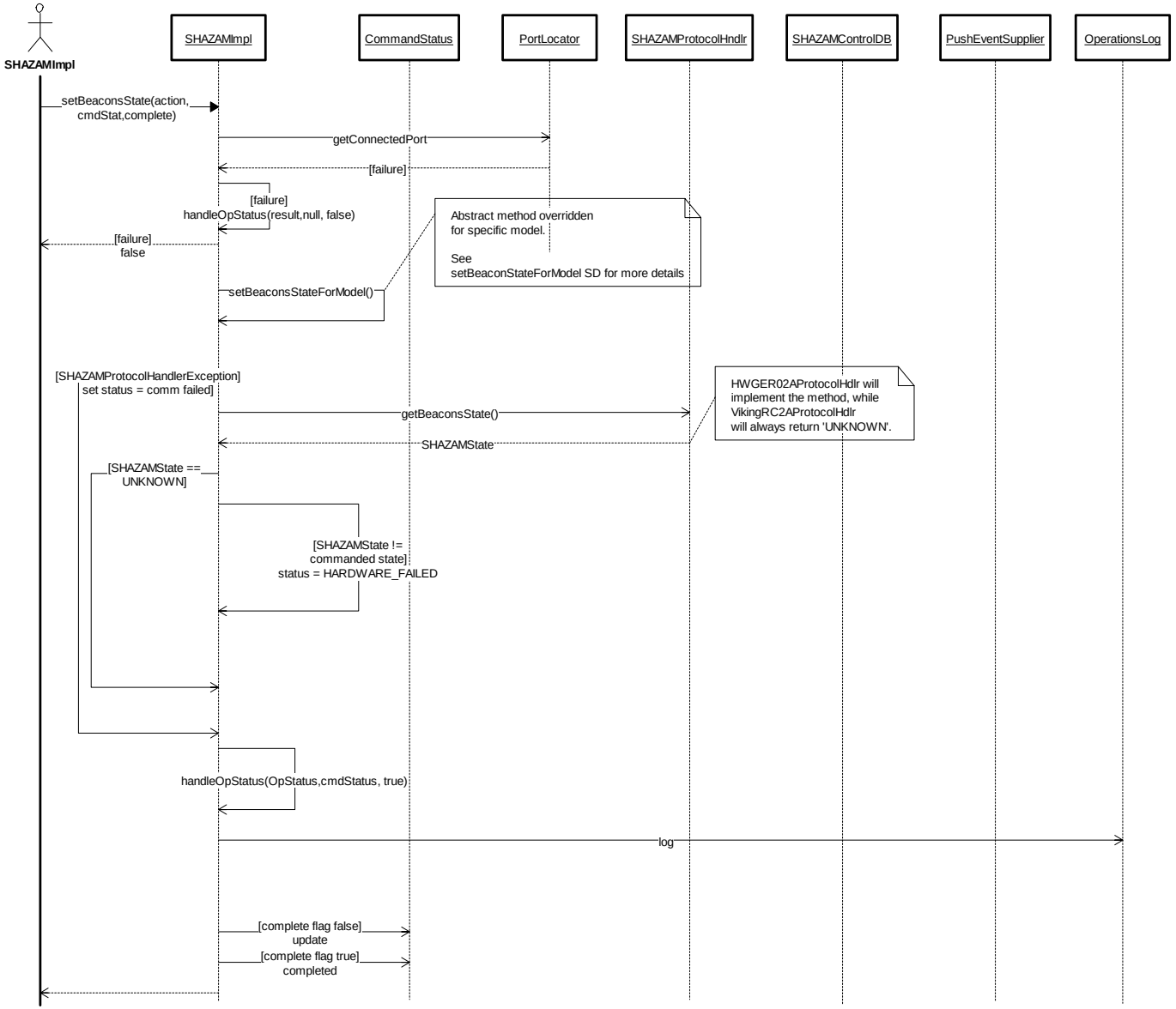


Figure 5-87. SHAZAMControlModule:setBeaconsState (Sequence Diagram)

5.7.2.11 SHAZAMControlModule:setConfiguration (Sequence Diagram)

A user with appropriate functional rights can set the configuration of a SHAZAM (VIKINGRC2A or HWGER02A) if it is in maintenance mode. Although this command does not currently require field communications, the asynchronous command pattern is used for consistency with other device commands and also to allow the code to easily adapt to a device type that supports configurable settings. When the command is executed, setConfigurationImpl stores configuration in memory. If it is communication parameters that have changed, a new PortLocator is created based on the modal. The new configuration is persisted to the database and an event is pushed onto the status event channel to notify others of the changes.

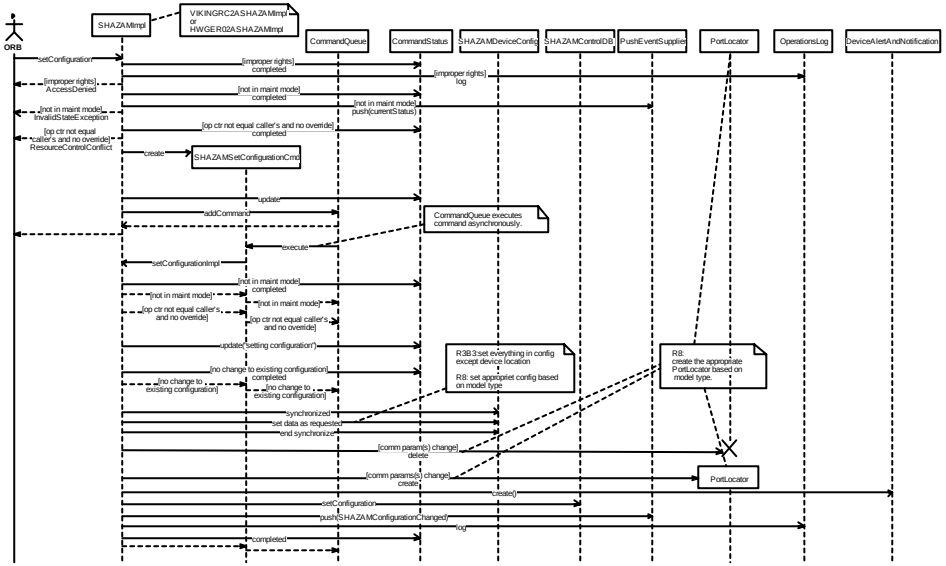


Figure 5-88. SHAZAMControlModule:setConfiguration (Sequence Diagram)

5.7.2.12 SHAZAMControlModule:updateNow (Sequence Diagram)

A user with appropriate functional rights can refresh SHAZAM now instead of waiting for the normal refresh cycle. This operation executes asynchronously. It returns to the caller after queuing the command for execution. A CommandStatus object may be used if the caller wishes to track the progress of the operation. After the beacon is contacted, a CurrentSHAZAMStatus type of SHAZAMEvent will be pushed through the event service SHAZAM channel.

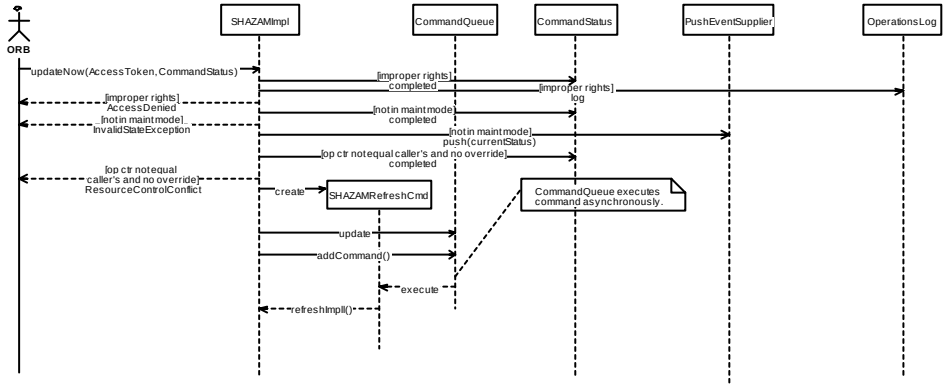


Figure 5-89. SHAZAMControlModule:updateNow (Sequence Diagram)

5.7.2.13 SHAZAMControlModule:remove (Sequence Diagram)

A user with the proper functional rights can remove an offline SHAZAM from the system. An token is generated using the TokenManipulator and the associated HAR (if there is one) is notified that the SHAZAM is to be removed. The SHAZAM object is withdrawn from the trader and disconnected from the ORB. The data for the SHAZAM is deleted from the database, including data that exists for the SHAZAM in the OBJECT_LOCATION and DEVICE_ALERT_NOTIFICATION tables. A message is pushed to the status event channel to allow the GUIs to remove the SHAZAM, and the command queue is shut down.

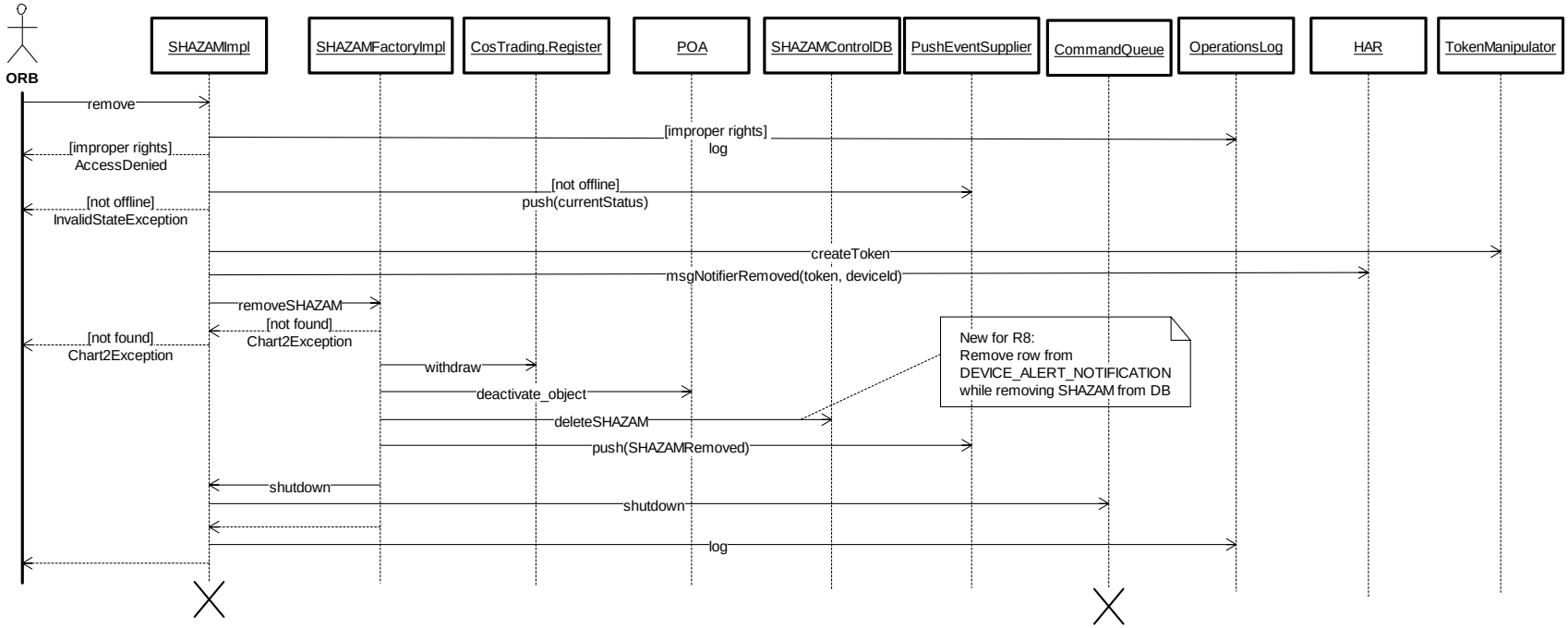


Figure 5-90 SHAZAMControlModule:remove (Sequence Diagram)

5.8 SHAZAM Protocols

5.8.1 Class Diagrams

5.8.1.1 SHAZAMProtocolsPkg (Class Diagram)

This class diagram shows the protocol handlers classes that are related to SHAZAM control.

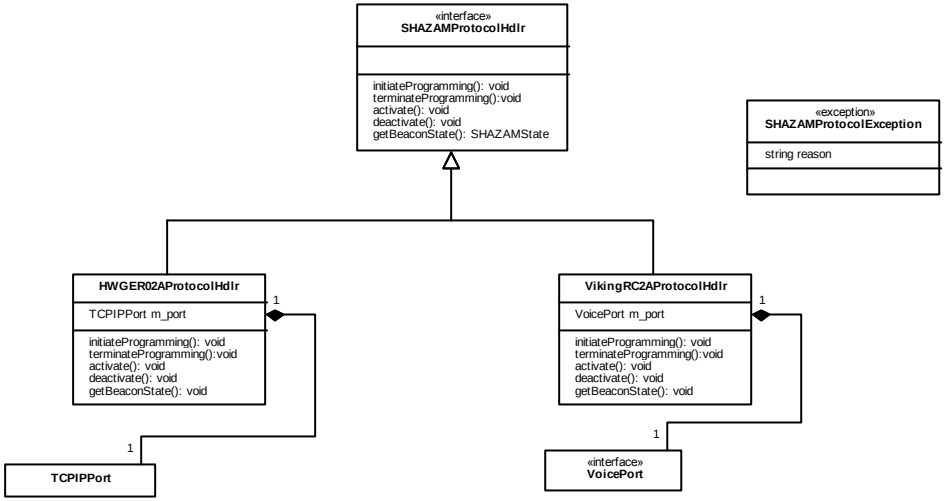


Figure 5-91. SHAZAMProtocolsPkg (Class Diagram)

5.8.1.1.1 HWGER02AProtocolHdlr (Class)

This protocol handler contains the protocol used to communicate with a HWG ER02A SHAZAM device.

5.8.1.1.2 SHAZAMProtocolException (Class)

This class represents an exception that is thrown by SHAZAM protocol classes when an unexpected error is encountered.

5.8.1.1.3 SHAZAMProtocolHdlr (Class)

This interface is used to provide methods for communicating with a SHAZAM device.

5.8.1.1.4 TCPIPPort (Class)

This class provides access to a TCP/IP port for device communications.

5.8.1.1.5 VikingRC2AProtocolHdlr (Class)

This protocol handler contains the protocol used to communicate with a Viking RC2A SHAZAM device.

5.8.1.1.6 VoicePort (Class)

A voice port provides access to a port on a telephony board. It provides methods to connect it to a destination phone number and perform send and receive operations while connected that result in DTMF or voice being sent across the telephone connection to or from the device.

5.9 chartlite.data.har

5.9.1 Class Diagrams

5.9.1.1 GUIHARDataClasses (Class Diagram)

This diagram shows classes used to store HAR related data in the GUI cache. New for R3B3 is the WebObjectLocation which is available in a WebHARConfig.

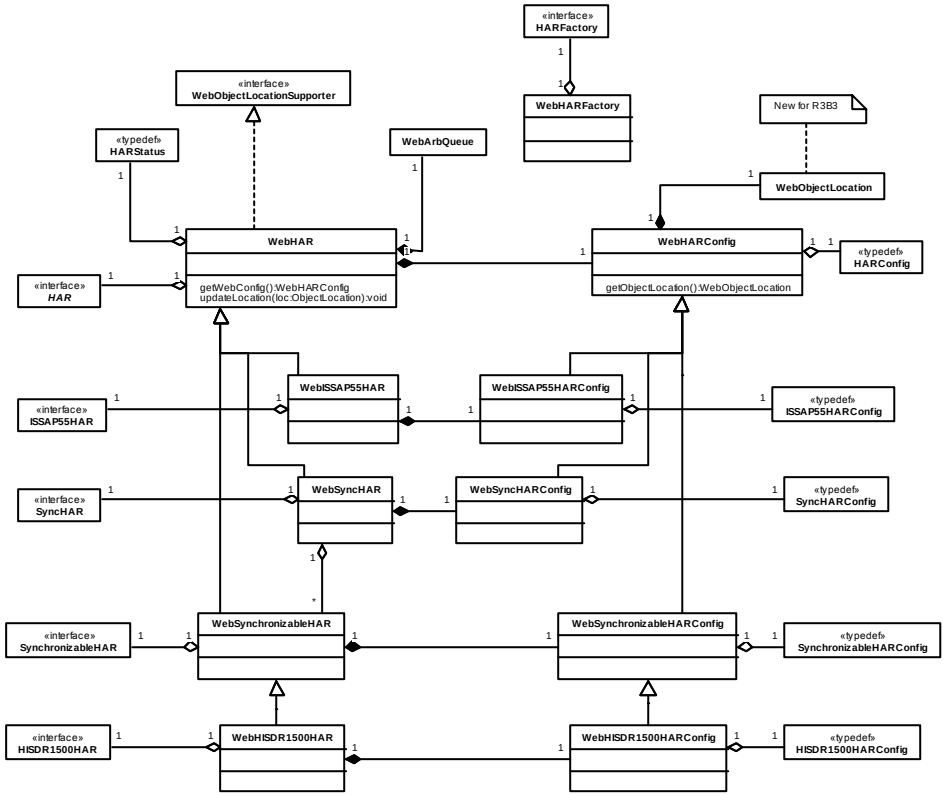


Figure 5-92. GUIHARDataClasses (Class Diagram)

5.9.1.1.1 HAR (Class)

This class is used to represent a Highway Advisory Radio (HAR) device. A HAR is used to broadcast traffic related information over a localized radio transmitter, making the information available to the traveler. This interface contains methods for getting and setting configuration, getting status, changing communications modes of a HAR, and manipulating and monitoring the HAR in maintenance and online modes.

5.9.1.1.2 HARConfig (Class)

This class holds data pertaining to a HAR device's configuration.

5.9.1.1.3 HARFactory (Class)

This CORBA interface allows new HAR objects to be added to the system. It also allows a requester to acquire a list of HAR objects under the domain of the specific HARFactory object.

5.9.1.1.4 HARStatus (Class)

This class (struct) contains data that indicates the current status of a HAR device. The data contained in this class is that status information which can be transmitted from the HAR to the client as necessary. This struct is also used to within the HAR Service to transmit data to/from the HARControlDB database interface class. (The HAR implementation also contains other private status data elements which are not elements of this class.)

5.9.1.1.5 HISDR1500HAR (Class)

This interface is implemented by objects that provide for the control of an HIS model DR1500 HAR.

5.9.1.1.6 HISDR1500HARConfig (Class)

This class holds configuration data for an HIS model DR1500 HAR.

5.9.1.1.7 ISSAP55HAR (Class)

This CORBA interface is implemented by objects that provide for the control of an ISS model AP55 HAR.

5.9.1.1.8 ISSAP55HARConfig (Class)

This class holds configuration data for an ISS model AP55 HAR

5.9.1.1.9 SyncHAR (Class)

This class is used to represent a synchronized Highway Advisory Radio (HAR) device. A synchronized HAR can have constituent HARs that it operates in a synchronized mode, allowing a continuous message to be delivered to the motorist as they travel out of range of one HAR and into the range of another.

5.9.1.1.10 SyncHARConfig (Class)

This class holds configuration data for a synchronized HAR.

5.9.1.1.11 SynchronizableHAR (Class)

This CORBA interface is implemented by objects that allow for control of HAR devices which can become constituents of a SyncHAR.

5.9.1.1.12 SynchronizableHARConfig (Class)

This class holds configuration for a HAR that can operate in a synchronized mode.

5.9.1.1.13 WebArbQueue (Class)

This class is a GUI wrapper for a CORBA ArbitrationQueue object.

5.9.1.1.14 WebHAR (Class)

This class is a GUI wrapper for a CORBA HAR object.

5.9.1.1.15 WebHARConfig (Class)

This class is a wrapper for a HARConfig object.

5.9.1.1.16 WebHARFactory (Class)

This class is a wrapper for a HARFactory used to store data pertaining to the HAR factory in the GUI cache.

5.9.1.1.17 WebHISDR1500HAR (Class)

This class is a GUI wrapper for a HISDR1500HAR CORBA object.

5.9.1.1.18 WebHISDR1500HARConfig (Class)

This class is a wrapper for a HISDR1500HARConfig object.

5.9.1.1.19 WebISSAP55HAR (Class)

This class is a GUI wrapper for a ISSAP55HAR CORBA object.

5.9.1.1.20 WebISSAP55HARConfig (Class)

This class is a wrapper for an ISSAP55HARConfig object.

5.9.1.1.21 WebObjectLocation (Class)

This class provides access to the ObjectLocation struct which contains information about the location of an object in the system.

5.9.1.1.22 WebObjectLocationSupporter (Class)

This interface allows common processing for objects supporting an ObjectLocation via the WebObjectLocation wrapper class.

5.9.1.1.23 WebSyncHAR (Class)

This class is a GUI wrapper for a SyncHAR CORBA object.

5.9.1.1.24 WebSyncHARConfig (Class)

This class is a wrapper for a SyncHARConfig object.

5.9.1.1.25 WebSynchronizableHAR (Class)

This class is a GUI wrapper for a SynchronizableHAR CORBA object.

5.9.1.1.26 WebSynchronizableHARConfig (Class)

This class is a wrapper for a SynchronizableHARConfig object.

5.10 chartlite.data.shazam

5.10.1 Class Diagrams

5.10.1.1 GUIShazamClasses (Class Diagram)

This diagram shows classes related to SHAZAMs that are used to store data pertaining to SHAZAMs in the GUI object cache.

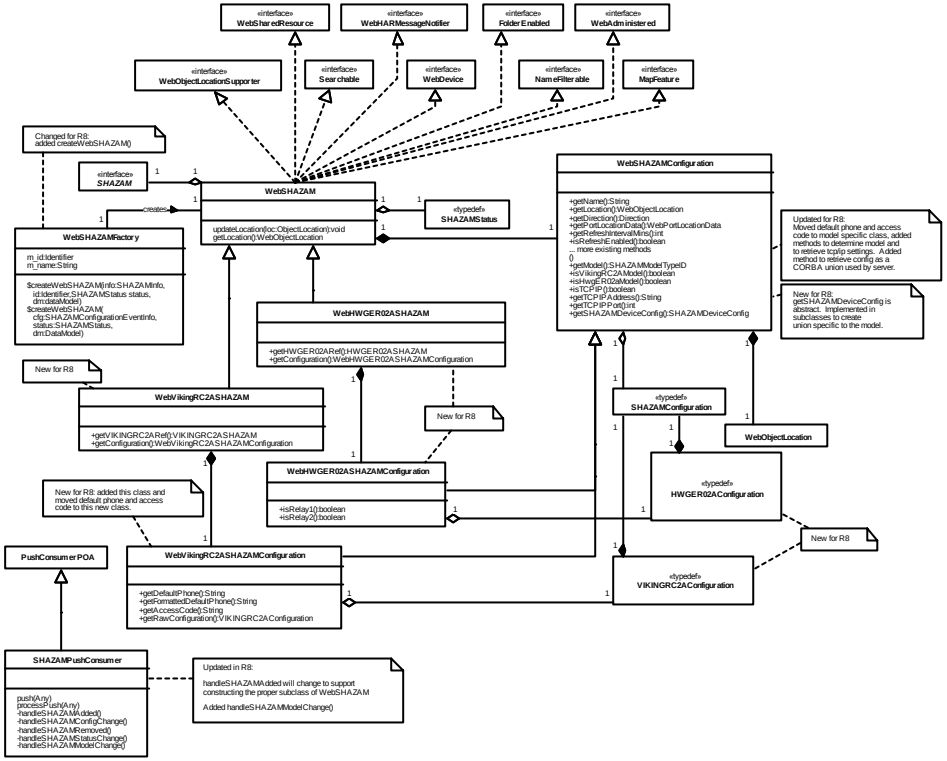


Figure 5-93. GUIShazamClasses (Class Diagram)

5.10.1.1.1 FolderEnabled (Class)

This interface provides access to information about an object that can be stored in a folder.

5.10.1.1.2 HWGER02AConfiguration (Class)

This class contains SHAZAMConfiguration plus data that is specific to a HWG ER02a SHAZAM.

5.10.1.1.3 MapFeature (Class)

This interface provides data necessary for displaying a feature on a map.

5.10.1.1.4 NameFilterable (Class)

This java interface is implemented by classes which can be filter by name within the ObjectCache. A NameFilter object is passed into the ObjectCache to select NameFilterable objects in the cache.

5.10.1.1.5 PushConsumerPOA (Class)

The CORBA base class for a push consumer, generated from IDL.

5.10.1.1.6 Searchable (Class)

This interface allows objects to be searched for via a substring search.

5.10.1.1.7 SHAZAM (Class)

This interface class is used to identify the common SHAZAM-specific methods which can be used to interface with a SHAZAM field device. It specifies methods for activating and deactivating the SHAZAM in maintenance mode, refreshing the SHAZAM (commanding the device to its last known status) and removing the SHAZAM. This interface is implemented by a SHAZAMImpl class, which uses a helper ProtocolHdlr class to perform the model specific protocol for device command and control.

5.10.1.1.8 SHAZAMConfiguration (Class)

This class contains data that specifies the configuration of a SHAZAM device. It is used to communicate configuration information to/from the database, and to/from the GUI clients. The GUI sends a SHAZAMConfiguration when creating a SHAZAM or modifying the configuration of an existing SHAZAM.Device Location member has been modified for R3B3.

Now it contains a detailed location information.

5.10.1.1.9 SHAZAMPushConsumer (Class)

This class handles SHAZAM related CORBA events that are pushed on a SHAZAM control event channel. The handleSHAZAMAdded method is updated in R8 to determine the model of the SHAZAM that has been added and to construct the proper WebSHAZAM derived object for that model. The handleSHAZAMModelChange method is new to R8 and handles SHAZAM model change events.

5.10.1.1.10 SHAZAMStatus (Class)

This class contains the current status of a SHAZAM device. This class is used to store status within the SHAZAM object, and is also used to communicate configuration information to/from the database, and to the GUI clients (one-way).

5.10.1.1.11 VIKINGRC2AConfiguration (Class)

This class contains SHAZAMConfiguration plus data that is specific to a Viking RC2A SHAZAM.

5.10.1.1.12 WebAdministered (Class)

This interface allows the implementing class to be administered via the trader console pages.

5.10.1.1.13 WebDevice (Class)

This interface contains common functionality for CHART devices.

5.10.1.1.14 WebHARMessageNotifier (Class)

This interface provides access to HAR notification capabilities for a device (DMS or SHAZAM) that is used to notify the public of a HAR message being broadcast.

5.10.1.1.15 WebHWGER02ASHAZAM (Class)

This class provides access to model specific configuration information that applies to the HWg ER02a SHAZAM.

5.10.1.1.16WebHWGER02ASHAZAMConfiguration (Class)

This class provides access to configuration information that is specific to an HWg ER02a SHAZAM.

5.10.1.1.17WebObjectLocation (Class)

This class provides access to the ObjectLocation struct which contains information about the location of an object in the system.

5.10.1.1.18WebObjectLocationSupporter (Class)

This interface allows common processing for objects supporting an ObjectLocation via the WebObjectLocation wrapper class.

5.10.1.1.19WebSharedResource (Class)

This interface is implemented by any GUI-side wrapper objects representing CHART shared resources in the system, corresponding to the SharedResource IDL interface.

5.10.1.1.20WebSHAZAM (Class)

This class is a wrapper for a SHAZAM CORBA object, used to cache data in the GUI object cache and provide access to the SHAZAM configuration and status data on web pages.

5.10.1.1.21WebSHAZAMConfiguration (Class)

This class is a wrapper for the SHAZAM configuration data, including its location. It can contain communication configuration for use of Telephony ports via port managers or for use of TCP/IP.

5.10.1.1.22WebSHAZAMFactory (Class)

This class is used to represent a SHAZAM factory available in the system. It has a static method that can be used to create an instance of WebSHAZAM derived object for a specific model of SHAZAM.

5.10.1.1.23WebVikingRC2ASHAZAM (Class)

This class provides access to data that is specific to a Viking RC2A SHAZAM. This SHAZAM is operated via a telephony

port and provides no responses to commands and does not support a status check.

5.10.1.1.24WebVikingRC2ASHAZAMConfiguration (Class)

This class provides access to configuration data that is specific to a Viking RC2A SHAZAM.

5.11 chartlite.servlet.har

5.11.1 Class Diagrams

5.11.1.1 GUIHARServletClasses (Class Diagram)

This diagram shows classes used by the servlet to process requests related to HAR devices.

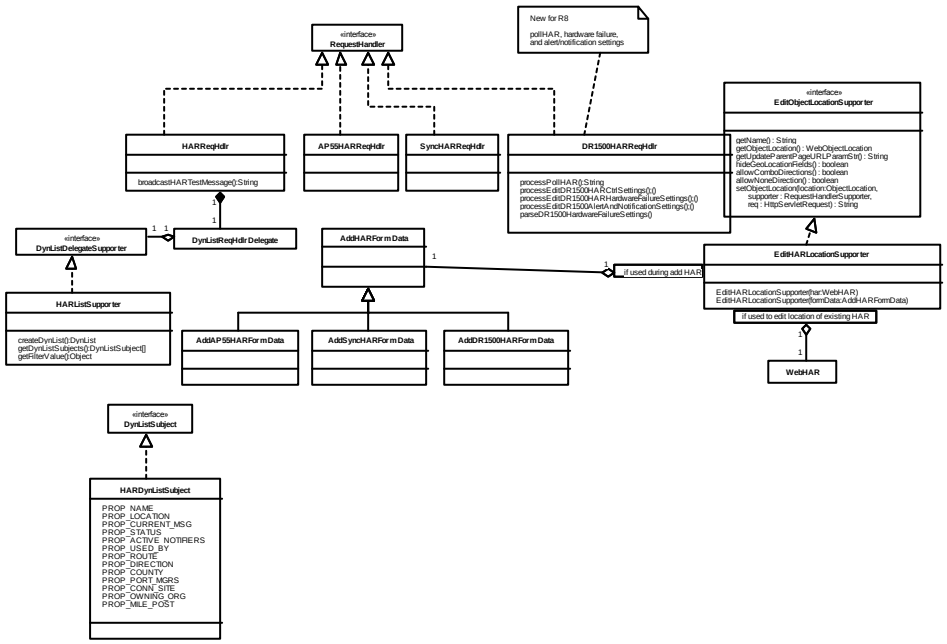


Figure 5-94. GUIHARServletClasses (Class Diagram)

5.11.1.1.1 AddAP55HARFormData (Class)

This class holds data specific to the AP55 HAR when adding an AP55 HAR to the system.

5.11.1.1.2 AddDR1500HARFormData (Class)

This class holds data specific to a DR1500 HAR when adding a DR1500 HAR to the system.

5.11.1.1.3 AddHARFormData (Class)

This class is used to store configuration data for a HAR during an Add operation, as the operation can require several web pages to complete and the data from each form must be stored between requests.

5.11.1.1.4 AddSyncHARFormData (Class)

This class holds data specific to a Sync HAR when adding a Sync HAR to the system.

5.11.1.1.5 AP55HARReqHdlr (Class)

This class handles requests that are specific to the AP55 HAR model.

5.11.1.1.6 DR1500HARReqHdlr (Class)

This class handles requests that are specific to the DR1500 HAR model. In R8 this request handler is updated to support polling of IP controlled DR1500 HARS.

5.11.1.1.7 DynListDelegateSupporter (Class)

This interface contains functionality to support the DynListReqHdlrDelegate

5.11.1.1.8 DynListReqHdlrDelegate (Class)

This class helps request handlers support dynamic lists. Requests to view, sort, or filter dynamic lists can be passed from a request handler to this class, provided the URL used for the requests contain parameters required by this class, such as the id of the list, the property name, and/or the filter value.

5.11.1.1.9 DynListSubject (Class)

This interface is implemented by classes that wish to be capable of being displayed in a dynamic list.

5.11.1.1.10 EditHARLocationSupporter (Class)

This class implements the EditObjectLocationSupporter interface for HARs. It can be constructed with an AddHARFormData object so it can be used during the Add HAR operation, in which case all location changes are stored in the AddHARFormData object. It can also be constructed using a WebHAR, for use when editing the location of an existing HAR. When this is done, the new location gets set into the actual HAR object (via a CORBA call).

5.11.1.1.11 EditObjectLocationSupporter (Class)

This interface provides functionality allowing the location data to be edited. (For example, the target of the edited location may be an existing object, or it may be a form data object for creating a new object).

5.11.1.1.12 HARDynListSubject (Class)

This class is a dyn list subject that holds a WebHAR. It also defines the property names for each column that will be shown in the list. The following columns are added in R3B3: route, direction, port managers, connection site, owning org, and mile post.

5.11.1.1.13 HARListSupporter (Class)

This class is a dyn list delegate supporter for the HAR list. It provides methods to create a dynamic list, get the subjects included in the list, and to get the value for a filter.

5.11.1.1.14 HARReqHdlr (Class)

This class handles requests that are valid for any HAR model.

5.11.1.1.15 RequestHandler (Class)

This interface specifies methods that are to be implemented by classes that are used to process requests.

5.11.1.1.16 SyncHARReqHdlr (Class)

This class handles requests that are specific to the Sync HAR model.

5.11.1.1.17WebHAR (Class)

This class is a GUI wrapper for a CORBA HAR object.

5.11.2 Sequence Diagrams

5.11.2.1 AddDR1500HARFormData:parseFormData (Sequence Diagram)

This diagram shows the processing that occurs when DR1500 HAR form parameters are read when a DR1500 is added. In R8, new parameters include polling, TCP/IP communications, hardware failure parameters, alerts, and notifications.

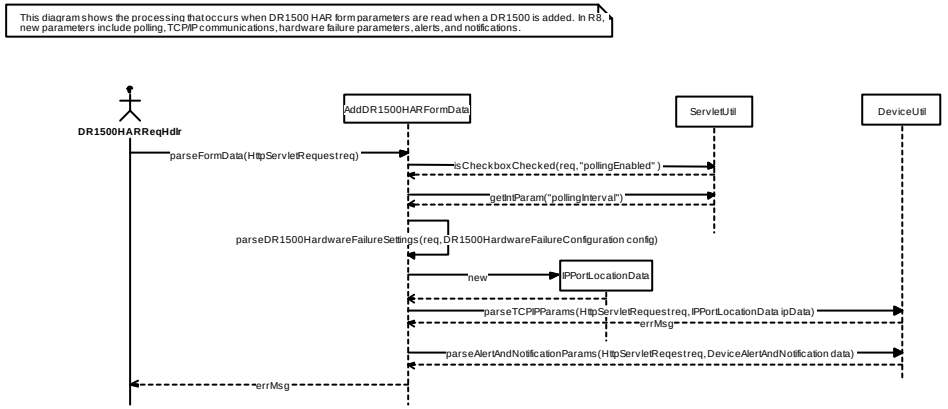


Figure 5-95. AddDR1500HARFormData:parseFormData (Sequence Diagram)

5.11.2.2 DR1500HARReqHdlr:processEditDR1500HARCtrlSettings (Sequence Diagram)

This diagram shows the processing that occurs when control communications settings are updated for a DR1500 HAR. Support for TCP/IP control is added in R8.

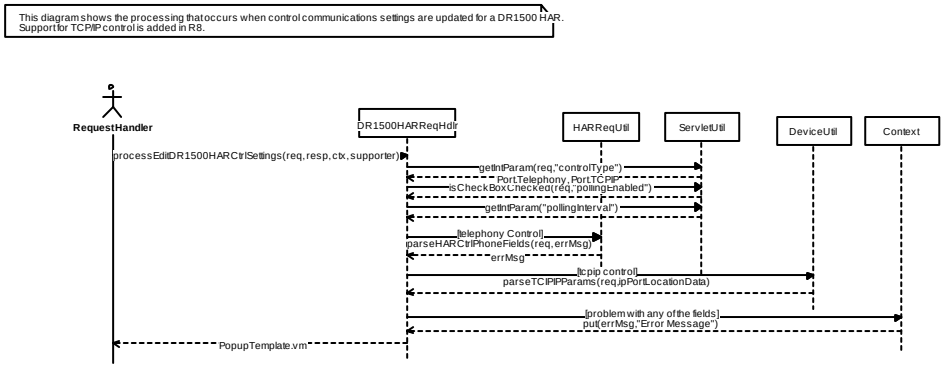


Figure 5-96. DR1500HARReqHdlr:processEditDR1500HARCtrlSettings (Sequence Diagram)

5.11.2.3 DR1500HARReqHdlr:processEditDR1500HARHardwareFailureSettings (Sequence Diagram)

This diagram shows the processing that occurs when updating the hardware failure settings for an IP controlled DR1500 HAR

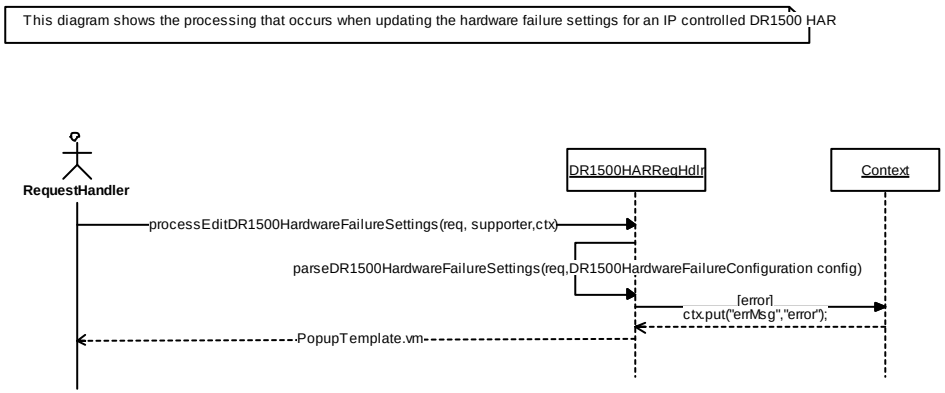


Figure 5-97. DR1500HARReqHdlr:processEditDR1500HARHardwareFailureSettings (Sequence Diagram)

5.11.2.4 DR1500HARReqHdlr:processPollHARNow (Sequence Diagram)

This diagram shows the processing that occurs when an immediate request to poll a har is received.

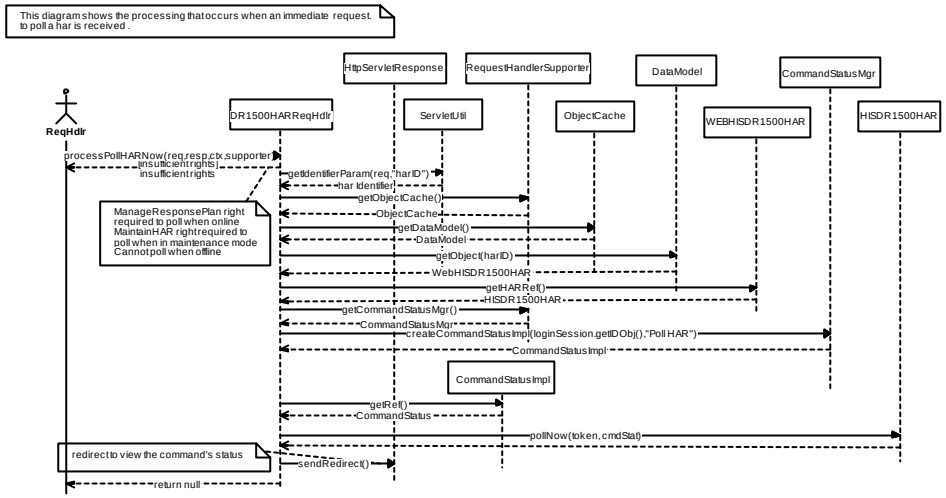


Figure 5-98. DR1500HARReqHdlr:processPollHARNow (Sequence Diagram)

5.11.2.5 DR1500HarReqHdlr:parseHardwareFailureSettings (Sequence Diagram)

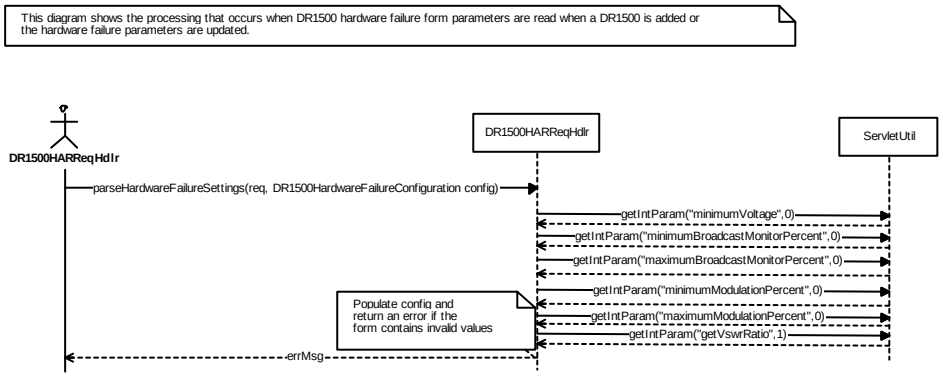


Figure 5-99. DR1500HarReqHdlr:parseHardwareFailureSettings (Sequence Diagram)

5.12 chartlite.servlet.shazam

5.12.1 Class Diagrams

5.12.1.1 GUISHAZAMServletClasses (Class Diagram)

This diagram shows classes used by the servlet to process requests related to SHAZAM devices.

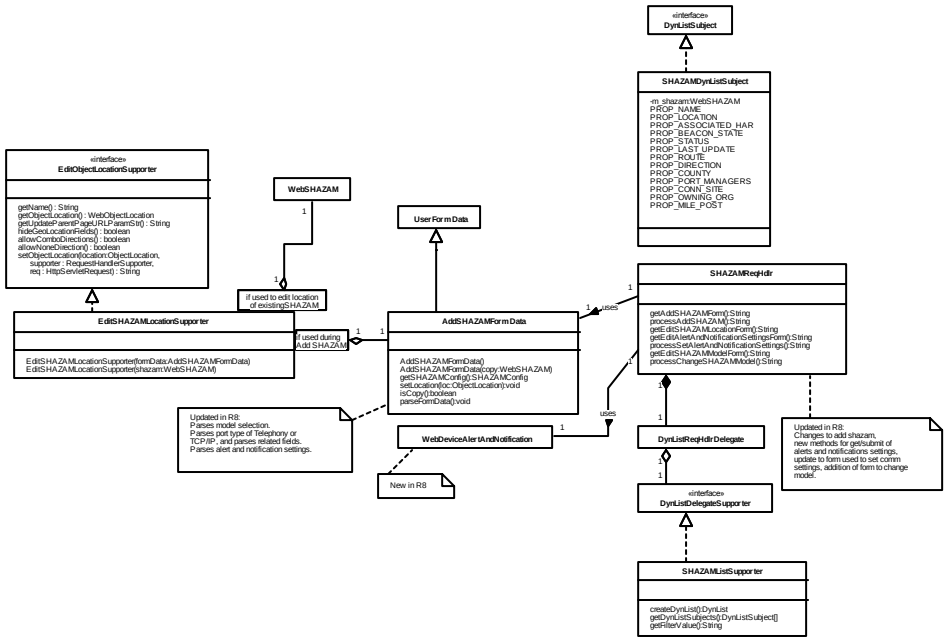


Figure 5-100. GUISHAZAMServletClasses (Class Diagram)

5.12.1.1.1 AddSHAZAMFormData (Class)

This class is used to store data from the Add SHAZAM form while interim pages are displayed (such as the edit location form).

5.12.1.1.2 DynListDelegateSupporter (Class)

This interface contains functionality to support the DynListReqHdlrDelegate

5.12.1.1.3 DynListReqHdlrDelegate (Class)

This class helps request handlers support dynamic lists. Requests to view, sort, or filter dynamic lists can be passed from a request handler to this class, provided the URL used for the requests contain parameters required by this class, such as the id of the list, the property name, and/or the filter value.

5.12.1.1.4 DynListSubject (Class)

This interface is implemented by classes that wish to be capable of being displayed in a dynamic list.

5.12.1.1.5 EditObjectLocationSupporter (Class)

This interface provides functionality allowing the location data to be edited. (For example, the target of the edited location may be an existing object, or it may be a form data object for creating a new object).

5.12.1.1.6 EditSHAZAMLocationSupporter (Class)

This class is used to support the generic edit location operation for SHAZAM devices.

5.12.1.1.7 SHAZAMDynListSubject (Class)

This class is a wrapper for a WebSHAZAM object that allows it to be displayed in a dynamic list.

5.12.1.1.8 SHAZAMListSupporter (Class)

This class is a DynListDelegateSupporter for the SHAZAM dynamic list. It provides SHAZAM specific functionality to the generic DynListReqHdlrDelegate.

5.12.1.1.9 SHAZAMReqHdlr (Class)

This class processes requests related to SHAZAM devices.

5.12.1.1.10 UserFormData (Class)

This class is used to store form data between requests while a user is editing a complex form, and provides convenience methods for parsing the values from the request.

5.12.1.1.11 WebDeviceAlertAndNotification (Class)

This class is a wrapper for a CORBA defined DeviceAlertAndNotification structure. It can be constructed from the CORBA struct and then be used to provide access to the data defined in that struct. It can also be constructed using parameters from an HTTP request and then used to extract a DeviceAlertAndNotification struct suitable for use when adding or editing a device.

5.12.1.1.12WebSHAZAM (Class)

This class is a wrapper for a SHAZAM CORBA object, used to cache data in the GUI object cache and provide access to the SHAZAM configuration and status data on web pages.

5.12.2 Sequence Diagrams

5.12.2.1 SHAZAMReqHdlr:getAddSHAZAMForm (Sequence Diagram)

This diagram shows the processing that is performed when the administrator chooses to add a SHAZAM to the system or create a new SHAZAM by copying an existing SHAZAM. This processing is also performed if an add or copy is already in progress and the Add form is being redisplayed after navigating away from the add form to set the location data. If this is the case, a formID parameter will be used to retrieve the form data object from the temp object store. If performing a copy, the WebSHAZAM being copied will be retrieved from the object cache and will be used to create an AddSHAZAMFormData object. This will have the effect of pre-populating the Add SHAZAM form with data from the SHAZAM being copied. If this is an Add SHAZAM operation that was not already in progress (neither formID nor shazamID is present in the request), a new empty AddSHAZAMFormData object is created. The form data is stored in the temp object store if not already stored there. Other objects needed for the form for select lists are obtained from the object cache and placed in the context. The Add SHAZAM form is then displayed to the user. Note: getAddSHAZAMForm exists prior to R8 and is modified in R8 to load the context to support the selection of alert op center and notification group for comm and hardware failures.

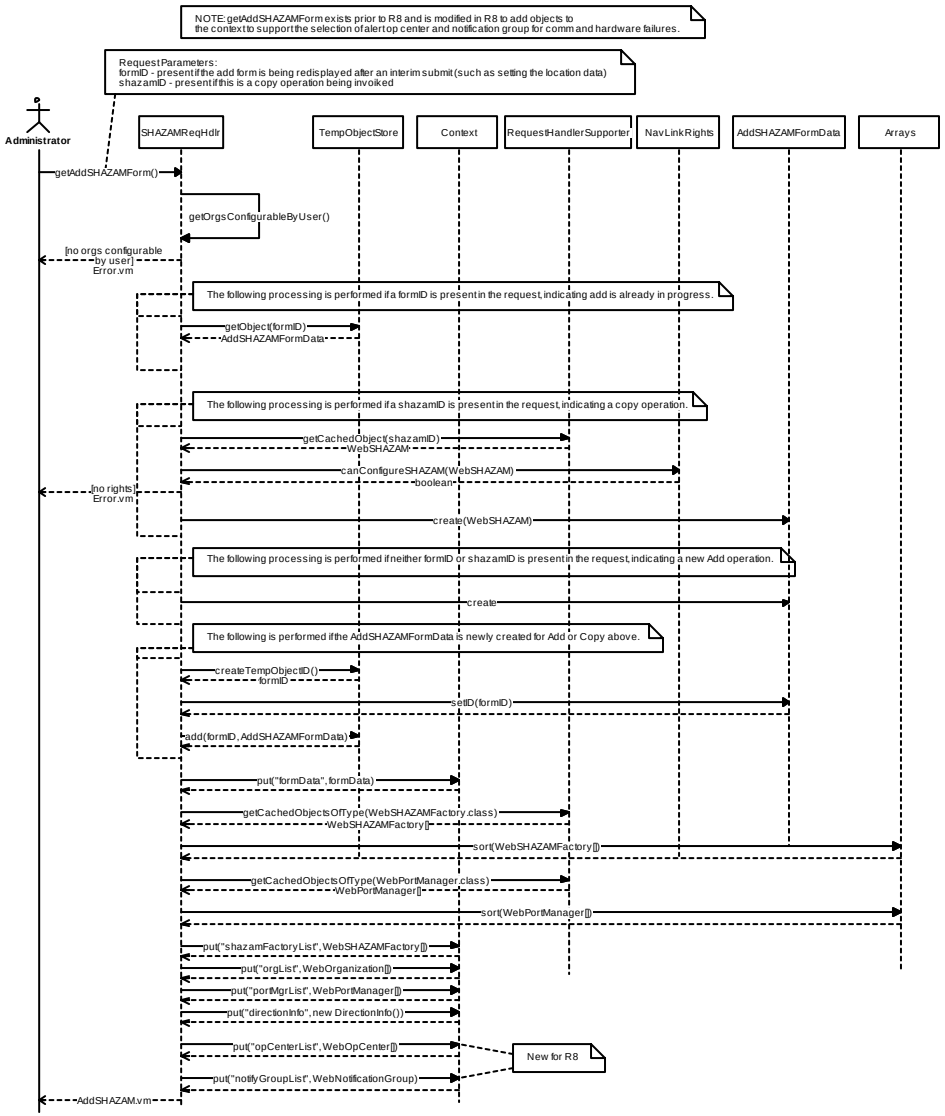


Figure 5-101. SHAZAMReqHdlr:getAddSHAZAMForm (Sequence Diagram)

5.12.2.2 SHAZAMReqHdlr:processAddSHAZAM (Sequence Diagram)

This diagram shows the processing that takes place when the user submits the Add SHAZAM Form. The formID is retreived from the request parameters and is used to retrieve the form data from the temp object store. The form data's populateFromRequest method is used to get all request parameters into the form data object, and its getSHAZAMConfig() method is used to populate a SHAZAMConfig object using the request parameters. The selected creation site (factory) is not part of the SHAZAMConfig, so it is retrieved from the form data separately. Any errors that are detected by the form data when creating the SHAZAMConfiguration or when attempting to get the selected factory ID cause the AddSHAZAM form to be redisplayed with an error message. If there were no errors, the SHAZAMConfiguration is passed to the SHAZAMFactory to create a new SHAZAM object. The new SHAZAM is then called to obtain its ID, status, and configuration to allow a WebSHAZAM object to be created and stored in the GUI cache. NOTE: processAddSHAZAM exists prior to R8 and is modified in R8 to support multiple SHAZAM models.

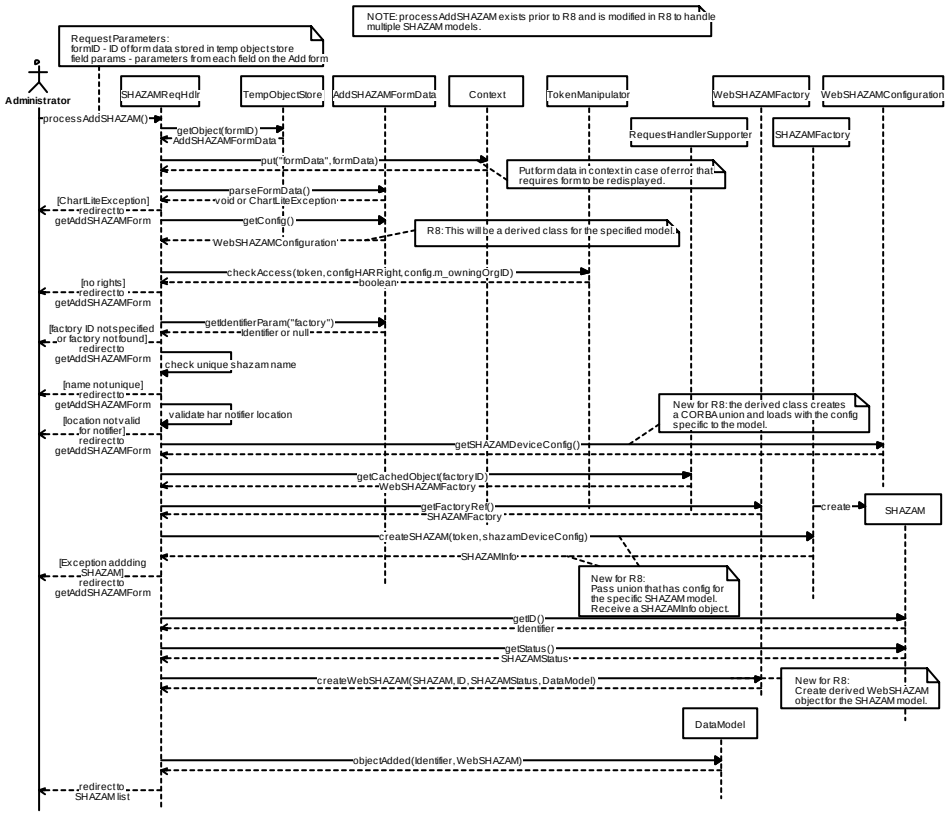


Figure 5-102. SHAZAMReqHdlr:processAddSHAZAM (Sequence Diagram)

5.12.2.3 SHAZAMReqHdlr:processChangeSHAZAMModel (Sequence Diagram)

This diagram shows the processing that is performed when the user submits the form used to change the SHAZAM model. The parameters are read and the WebSHAZAM object for the specified SHAZAM is retrieved from the data model. The user's rights are checked to make sure they have permission to modify the configuration for the SHAZAM. The SHAZAM status is checked to make sure the SHAZAM is offline. The SHAZAM CORBA object reference is obtained and a command

status object is created. The SHAZAM CORBA object is called to change the model; status of the operation will be reported back to the user via the CommandStatus object. The user's browser is then redirected to the command status page where they can monitor the progress of the operation.

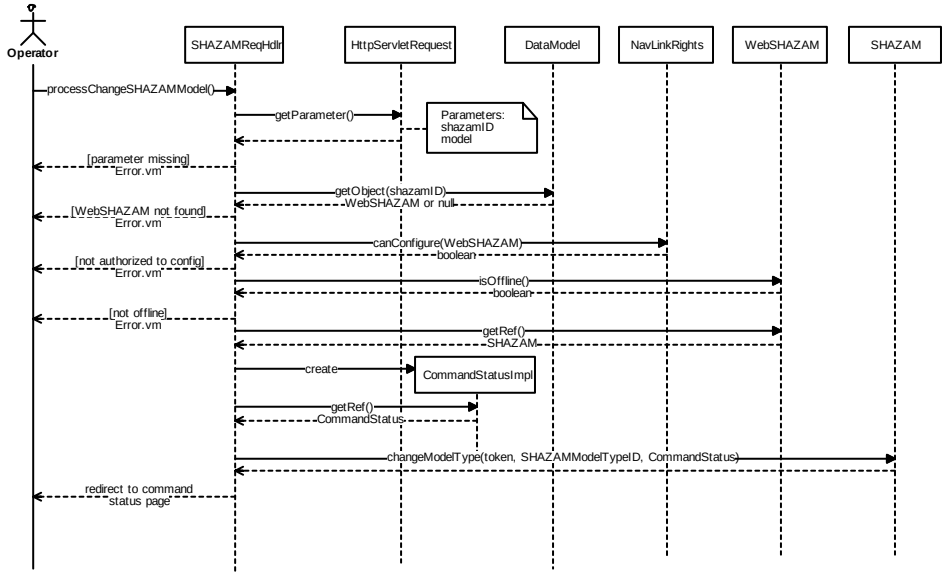


Figure 5-103. SHAZAMReqHdlr:processChangeSHAZAMModel (Sequence Diagram)

5.12.2.4 SHAZAMReqHdlr:processEditCommsConfig (Sequence Diagram)

This diagram shows the processing performed when the user submits the form used to edit the communication settings for the SHAZAM. The parameters are read from the request with the set of required parameters dependent on the port type specified. If parameters required for the selected port type are missing or invalid an error is returned. The SHAZAM ID is used to find the WebSHAZAM wrapper object in the GUI cache. If not found an error is returned. The user rights are checked to validate that the user has the right to change the configuration of the SHAZAM; if not an error message is returned. The configuration of the SHAZAM is retrieved from the WebSHAZAM and the model of the SHAZAM is checked. Only TCP/IP

communications are supported for HWG ER02A, and only Telephony communications are supported for Viking RC2A. The model specific CORBA reference is obtained and called to retrieve the model specific configuration from the server. The parameters from the request are loaded into the appropriate members of the configuration object. A CommandStatus object is created, and the model specific CORBA reference's setConfiguration method is called. The user's browser is then redirected to the view command status page where they can monitor the progress of the setConfiguration operation.

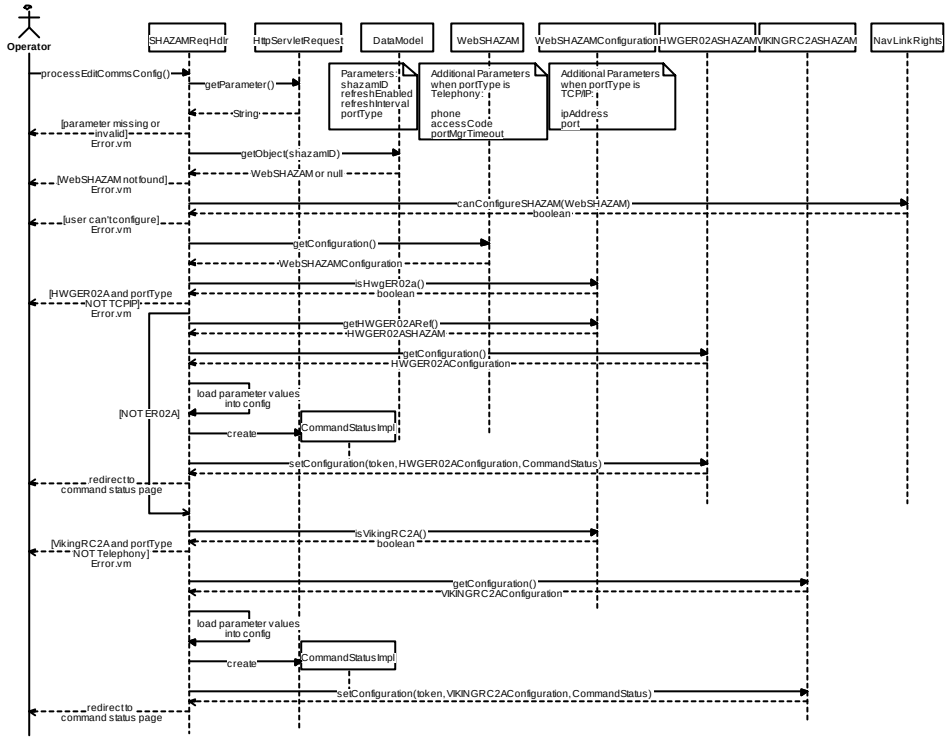


Figure 5-104. SHAZAMReqHdlr:processEditCommsConfig (Sequence Diagram)

5.12.2.5 SHAZAMReqHdlr:processSetAlertAndNotificationSettings (Sequence Diagram)

This diagram shows the processing that is performed when the user submits the form used to set the alert and notification settings for a SHAZAM. The SHAZAM is retrieved from the GUI cache and a check is made to verify the user is authorized to configure the SHAZAM. The form parameters are retreived and parsed by wrapping the HttpServletRequest into an HttpRequestParameterSupplier and passing that supplier to the constructor of the WebDeviceAlertandNotification class. The existing SHAZAM configuration is retrieved from the server and the DeviceAlertAndNotification object in the configuration is replaced with the object parsed from the request parameters. A CommandStatus object is created and the SHAZAM's setConfiguration() method is called to set the configuration asynchronously. The user's browser is redirected to the command status page where they can monitor the progress of the command. This process supports multiple SHAZAM models - the proper SHAZAM derived object reference is used to retrieve and set the model specific configuration.

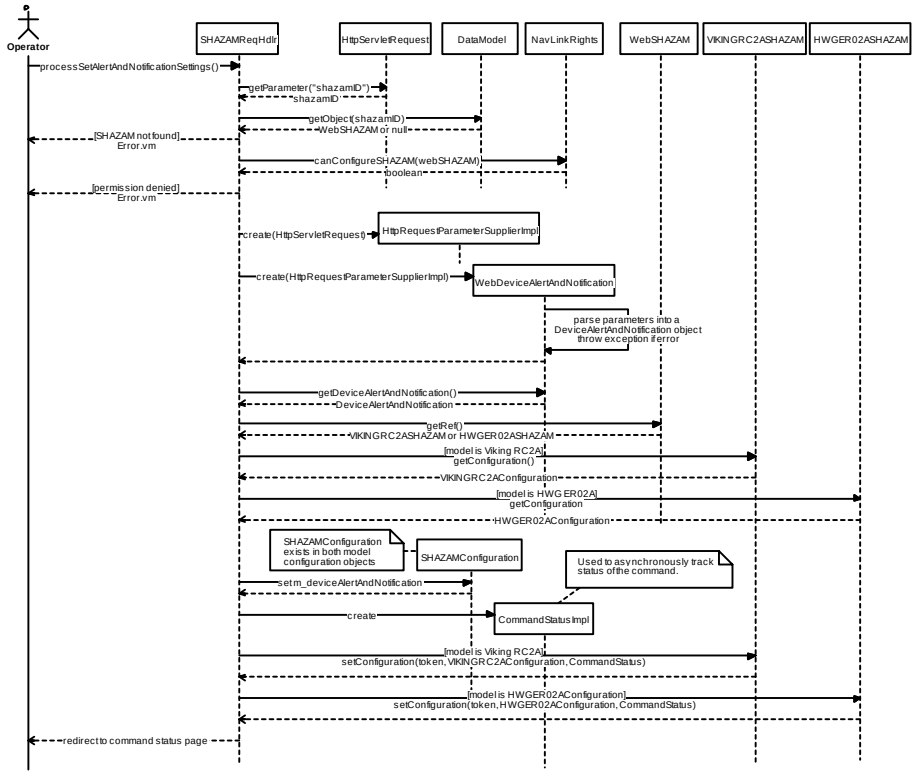


Figure 5-105. SHAZAMReqHdlr:processSetAlertAndNotificationSettings (Sequence Diagram)

5.13 chartlite utilities

5.13.1 Class Diagrams

5.13.1.1 chartlite.util_classes (Class Diagram)

This diagram shows utility classes used in the CHART GUI servlet.

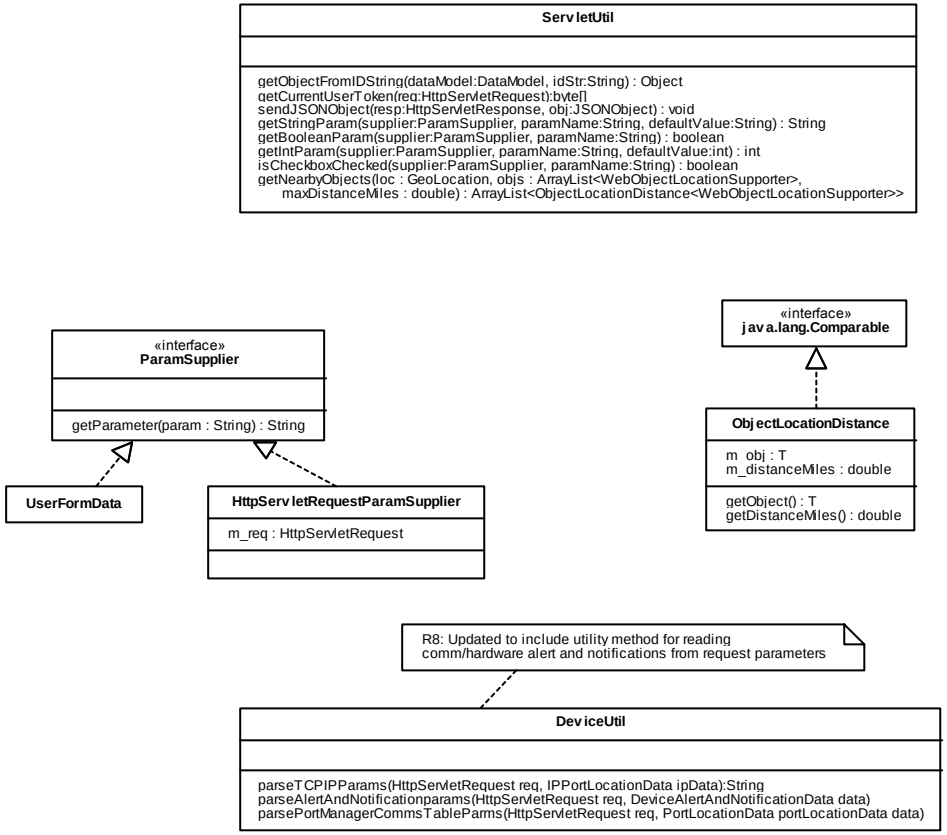


Figure 5-106. chartlite.util_classes (Class Diagram)

5.13.1.1.1 DeviceUtil (Class)

This utility class contains non device specific utility methods.

5.13.1.1.2 HttpServletRequestParamSupplier (Class)

This class implements the ParamSupplier interface to provide parameters from an HttpServletRequest object.

5.13.1.1.3 java.lang.Comparable (Class)

This interface allows two objects to be compared for the purposes of sorting.

5.13.1.1.4 ObjectLocationDistance (Class)

This class stores an object and a calculated distance to another point. It is used to avoid re-calculating the distance multiple times.

5.13.1.1.5 ParamSupplier (Class)

This interface allows parameter values to be retrieved. It was added to handle parameters supplied by a HttpServletRequest and form data using common code.

5.13.1.1.6 ServletUtil (Class)

This class provides static utility methods useful to request handlers in the servlet.

5.13.1.1.7 UserFormData (Class)

This class is used to store form data between requests while a user is editing a complex form, and provides convenience methods for parsing the values from the request.

6 Deprecated Functionalities

No CHART functionality is being deprecated with the introduction of R8.

7 Exporter Changes

The following table shows data fields being added in R8 and indicates which will be exported and which will not be exported.

HAR	TCP_HOST	WILL be Exported	WILL NOT be exported
	TCP_PORT		X
	MIN_DC_VOLTAGE		X
	MIN_BROADCAST_MONITOR_PCT		X
	MAX_BROADCAST_MONITOR_PCT		X
	MIN_MODULATION_PCT		X
	MAX_MODULATION_PCT		X
	MAX_VSWR		X
	LAST_SETUP_TIME		X
	LAST_STATUS_MISMATCH_TIME		X
	POWER_STATUS		X
	DC_VOLTAGE		X
	BROADCAST_MONITOR_PCT		X
	HAR_MODE		X
	HAR_SUB_MODE		X

	HAR_SYNC_MODE		X
	XMIT_SET_POWER		X
	XMIT_FORWARD_POWER		X
	XMIT_REFLECTED_POWER		X
	XMIT_VSWR		X
	XMIT_MODULATION_PCT		X
	DCC_VERSION_INFO		X
	HAR_VERSION_INFO		X
	HAR_TIMESTAMP		X
	COMMFAIL_ALERT_CENTER_ID		X
	HWFAIL_ALERT_CENTER_ID		X
	COMMFAIL_NOTIF_GROUP_ID		X
	COMMFAIL_NOTIF_GROUP_NAME		X
	HWFAIL_NOTIF_GROUP_ID		X
	HWFAIL_NOTIF_GROUP_NAME		X
SHAZAM	TCP_HOST		X
	TCP_PORT		X
	RELAY_NUMBER		X
	BEACON_STATE_ACTUAL		X
	COMMFAIL_ALERT_CENTER_ID		X
	HWFAIL_ALERT_CENTER_ID		X
	COMMFAIL_NOTIF_GROUP_ID		X

	COMMFAIL_NOTIF_GROUP_NAME		X
	HWFAIL_NOTIF_GROUP_ID		X
	HWFAIL_NOTIF_GROUP_NAME		X

8Mapping To Requirements

The following table shows how the requirements in the CHART R8 Requirements document map to design elements contained in this design.

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1	ADMINISTER SYSTEMS AND EQUIPMENT		N/A	N/A (Heading)
SR1.5	INSTALL AND MAINTAIN DEVICES		N/A	N/A (Heading)
SR1.5.2	PUT EQUIPMENT/ DEVICES ON-LINE		N/A	N/A (Heading)
SR1.5.2.1	The system shall allow the user with appropriate rights to select (or modify) the equipment device parameters.		Set HAR Configuration Set SHAZAM Configuration	N/A (General)
SR1.5.2.1.5	The system shall allow a suitably privileged user to add a new device which communicates via an implemented protocol.		Add HAR Add SHAZAM	SHAZAMReqHdlr:processAddSHAZAM SD SHAZAMControlModule.createSHAZAM SD
SR1.5.2.1.5.2	[MOVED TO 1.5.2.1.5.11.1] The system shall support HAR communication via a Field Management Server that implements Telephony communications mediums.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.2.1	[MOVED TO 1.5.2.1.5.11.3] The system shall support sending messages to at least four HARs simultaneously. Each individual constituent HAR within a Synchronized HAR will count as one HAR against this total.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.5.2.1.1	[MOVED TO 1.5.2.1.5.11.3.1] A Synchronized HAR shall be comprised of individual constituent HARs that play messages in such a way that the messages on each individual constituent HAR play at exactly the same time.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.2.1.1.1	[MOVED TO 1.5.2.1.5.11.3.1.1] The system shall support HAR message synchronization for HIS DR1500 HARs.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.2.1.1.2	[MOVED TO 1.5.2.1.5.11.3.1.2] Individual constituent HARs within a Synchronized HAR may be designated as active or inactive.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.2.1.1.2.1	[MOVED TO 1.5.2.1.5.11.3.1.2.1] An inactive individual constituent HAR within a Synchronized HAR shall not broadcast any message.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.2.1.1.2.2	[MOVED TO 1.5.2.1.5.11.3.1.2.2] Each active individual constituent HAR within a Synchronized HAR shall always broadcast an identical message in a synchronized manner.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.2.2	[MOVED TO 1.5.2.1.5.11.4] If a HAR has Notifiers (SHAZAMs or DMSs acting as SHAZAMs) associated with it and a message that requires Notifiers on is sent to it, the specified Notifiers shall be turned on only after a message has been activated on the HAR .	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.2.3	[MOVED TO 1.5.2.1.5.11.5] If a HAR has active Notifiers (SHAZAMs or DMSs acting as SHAZAMs), the Notifiers shall be turned off before the current message is deactivated.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.5.3	[MOVED TO 1.5.2.1.5.12.1] The system shall support SHAZAM communication via a Field Management Server that implements Telephony communications mediums.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.5.11	The system shall support HAR communications.	HAR/SHAZAM	Set HAR Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData
SR1.5.2.1.5.11.1	The system shall support HAR communication via a Field Management Server that implements Telephony communications mediums.	HAR/SHAZAM	Set Telephony Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData AP55AndDR1500HARCommand.getDTMFCommand SD
SR1.5.2.1.5.11.2	The system shall support HAR communications via a TCP/IP communications medium.	HAR/SHAZAM	Set TCPIP Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData HARProtocolsPkg/HARProtocolsPkg.cad HARProtocolsPkg/HARProtocolsHdlrDispatch.cad HISDR1500ProtocolHdlr:sendSerialDataToHAR SD AP55andDR1500HARCommand.getByteCommand SD HISDR1500ProtocolHdlr:parseByteResponse SD ByteUtil:asciiBytesToInt SD ByteUtil:intToAsciiBytes SD
SR1.5.2.1.5.11.3	The system shall support sending messages to at least four HARs simultaneously. Each individual constituent HAR within a Synchronized HAR will count as one HAR against this total.	HAR/SHAZAM	Set HAR Message	HISDR1500ProtocolHdlr:BroadcastSlots SD HISDR1500ProtocolHdlr:recordMessage SD HISDR1500ProtocolHdlr:reclaimMemory
SR1.5.2.1.5.11.3.1	A Synchronized HAR shall be comprised of individual constituent HARs that play messages in such a way that the messages on each individual constituent HAR play at exactly the same time.	HAR/SHAZAM	Set HAR Message	HISDR1500ProtocolHdlr:BroadcastSlots SD HISDR1500ProtocolHdlr:recordMessage SD HISDR1500ProtocolHdlr:reclaimMemory
SR1.5.2.1.5.11.3.1.1	The system shall support HAR message synchronization for HIS DR1500 HARs.	HAR/SHAZAM	Set HAR Message	HISDR1500ProtocolHdlr:BroadcastSlots SD HISDR1500ProtocolHdlr:recordMessage SD HISDR1500ProtocolHdlr:reclaimMemory

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.5.11.3.1.2	Individual constituent HARs within a Synchronized HAR may be designated as active or inactive.	HAR/SHAZAM	Set HAR Message	Use Case Only
SR1.5.2.1.5.11.3.1.2.1	An inactive individual constituent HAR within a Synchronized HAR shall not broadcast any message.	HAR/SHAZAM	Set HAR Message	Use Case Only
SR1.5.2.1.5.11.3.1.2.2	Each active individual constituent HAR within a Synchronized HAR shall always broadcast an identical message in a synchronized manner.	HAR/SHAZAM	Set HAR Message	Use Case Only
SR1.5.2.1.5.11.4	If a HAR has Notifiers (SHAZAMs or DMSs acting as SHAZAMs) associated with it and a message that requires Notifiers on is sent to it, the specified Notifiers shall be turned on only after a message has been activated on the HAR.	HAR/SHAZAM	Set HAR Message	Use Case Only
SR1.5.2.1.5.11.5	If a HAR has active Notifiers (SHAZAMs or DMSs acting as SHAZAMs), the Notifiers shall be turned off before the current message is deactivated.	HAR/SHAZAM	Blank HAR	Use case Only
SR1.5.2.1.5.12	The system shall support SHAZAM communications.	HAR/SHAZAM	Set SHAZAM Communication Settings	Use Case Only
SR1.5.2.1.5.12.1	The system shall support SHAZAM communication via a Field Management Server that implements Telephony communications mediums.	HAR/SHAZAM	Set SHAZAM Communication Settings	SHAZAMReqHdlr:processEditCommsConfig SD SHAZAMControlModule.setConfiguration SD
SR1.5.2.1.5.12.2	The system shall support SHAZAM communications via a TCP/IP communications medium.	HAR/SHAZAM	Set SHAZAM Communication Settings	SHAZAMReqHdlr:processEditCommsConfig SD SHAZAMControlModule.setConfiguration SD
SR1.5.2.1.8	The system shall support configuration parameters for HAR devices.	HAR/SHAZAM	Set HAR Configuration	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings SD AddDR1500HARFormData:parseFormData SD DR1500HARReqHdlr:parseHardwareFailureSettings SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.8.13	Specify HIS DR1500 HAR Configuration	HAR/SHAZAM	Set HAR Configuration	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings SD AddDR1500HARFormData:parseFormData SD DR1500HARReqHdlr:parseHardwareFailureSettings SD SHAZAMContorlModule.setConfiguration SD
SR1.5.2.1.8.13.4	[CHANGED AND MOVED TO 1.5.2.1.8.13.18.1.3] The system shall allow the user to specify the access code for a HIS DR1500 HAR.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.8.13.5	[CHANGED AND MOVED TO 1.5.2.1.8.13.18.2.1] The system shall allow the user to specify the default phone number of a HIS DR1500 HAR's control line (10, 11, or 12 digits).	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.8.13.6	[CHANGED AND MOVED TO 1.5.2.1.8.13.18.2.2] The system shall allow the user to specify the port manager configuration for a HIS DR1500 HAR's control line, as defined in the Specify Port Manager Configuration requirements.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.8.13.7	[CHANGED AND MOVED TO 1.5.2.1.8.13.18.1] The system shall indicate to the user that a Telephony-type port will be used when communicating with a HIS DR1500 HAR's control line.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.8.13.18	The system shall allow a suitably privileged user to specify the port type to be used to communicate with a HIS DR1500 HAR.	HAR/SHAZAM	Set HAR Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData
SR1.5.2.1.8.13.18.1	The system shall allow the user to specify that a Telephony port shall be used to communicate with a HIS DR1500 HAR.	HAR/SHAZAM	Set HAR Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.8.13.18.1.1	The system shall allow the user to specify the default phone number of a HIS DR1500 HAR's control line (10, 11, or 12 digits) when the configuration indicates a Telephony port is to be used to communicate with the HAR.	HAR/SHAZAM	Set Telephony Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData
SR1.5.2.1.8.13.18.1.2	The system shall allow the user to specify the port manager configuration for a HIS DR1500 HAR's control line, as defined in the Specify Port Manager Configuration requirements, when the HAR configuration specifies a Telephony port is used to communicate with the HAR.	HAR/SHAZAM	Set Telephony Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData
SR1.5.2.1.8.13.18.1.3	The system shall allow the user to specify the access code for a HIS DR1500 HAR that is set to communicate via a Telephony port.	HAR/SHAZAM	Set HAR Communication Settings	HARControlModule:fmsGetConnectedPort SD
SR1.5.2.1.8.13.18.2	The system shall allow the user to specify that a TCP/IP port shall be used to communicate with a HIS DR1500 HAR.	HAR/SHAZAM	Set HAR Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData
SR1.5.2.1.8.13.18.2.1	The system shall allow the user to specify the IP address and port of a HIS DR1500 HAR when its configuration indicates a TCP/IP port is to be used to communicate with the HAR.	HAR/SHAZAM	Set TCPIP Communication Settings	DR1500HARReqHdlr:processEditDR1500HARCtrlSettings AddDR1500HARFormData:parseFormData
SR1.5.2.1.8.13.18.2.2	The system shall allow the user to specify if a HIS DR1500 HAR is to be automatically polled when the configuration specifies that a TCP/IP port is to be used to communicate with the HAR.	HAR/SHAZAM	Set HAR Configuration	AddDR1500HARFormData:parseFormData
SR1.5.2.1.8.13.18.2.2.1	The system shall allow the Polling Interval to be specified if automatic polling is enabled for a HIS DR1500 HAR.	HAR/SHAZAM	Set HAR Configuration	AddDR1500HARFormData:parseFormData

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.8.13.18.2.2.2	The default polling rate for a HIS DR1500 HAR shall be 5 minutes.	HAR/SHAZAM	Set HAR Configuration	Use Case Only
SR1.5.2.1.8.13.18.2.4	The system shall default the TCP/IP port to 200 when adding a HIS DR1500 HAR to the system. (The user can override this default)	HAR/SHAZAM	Set HAR Configuration	Use Case Only
SR1.5.2.1.8.13.20	The system shall allow a suitably privileged user to specify thresholds used to detect hardware failures when the configuration specifies that a TCP/IP port is to be used to communicate with a HIS DR1500 HAR.	HAR/SHAZAM	Set DR1500 Hardware Failure Detection Settings	DR1500HARReqHdlr:parseHardwareFailureSettings DR1500HARReqHdlr:processHardwareFailureSettings AddDR1500HARFormData:parseFormData
SR1.5.2.1.8.13.20.1	The system shall allow the user to specify the minimum voltage value, used to determine if the HIS DR1500 HAR is hardware failed.	HAR/SHAZAM	Set DR1500 Hardware Failure Detection Settings	DR1500HARReqHdlr:parseHardwareFailureSettings DR1500HARReqHdlr:processHardwareFailureSettings
SR1.5.2.1.8.13.20.4	The system shall allow the user to specify a maximum voltage standing wave ratio (VSWR), used to determine if the HIS DR1500 HAR is hardware failed.	HAR/SHAZAM	Set DR1500 Hardware Failure Detection Settings	DR1500HARReqHdlr:parseHardwareFailureSettings DR1500HARReqHdlr:processHardwareFailureSettings
SR1.5.2.1.8.13.21	The system shall allow a suitably privileged user to specify alert and notification settings for a HAR.	HAR/SHAZAM	Set HAR Configuration	AlertAndNotificationHelper:notifyAndAlert SD
SR1.5.2.1.8.13.21.1	The system shall support setting a responsible Center for a HAR which is to receive the Device Failure Alert when the device goes into hardware failure.	HAR/SHAZAM	Set Alert and Notification Settings	AlertAndNotificationHelper:notifyAndAlert SD
SR1.5.2.1.8.13.21.2	The system shall support setting a responsible Center for a HAR which is to receive the Device Failure Alert when the device goes into communication failure.	HAR/SHAZAM	Set Alert and Notification Settings	AlertAndNotificationHelper:notifyAndAlert SD
SR1.5.2.1.8.13.21.3	The system shall support setting a notification group to receive HAR communication failure notification messages.	HAR/SHAZAM	Set Alert and Notification Settings	AlertAndNotificationHelper:notifyAndAlert SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.8.13.21.4	The system shall support setting a notification group to receive HAR hardware failure notification messages.	HAR/SHAZAM	Set Alert and Notification Settings	AlertAndNotificationHelper:notifyAndAlert SD
SR1.5.2.1.8.13.21.5	The system shall support setting an operations center or notification group to "None" to disable the sending of alerts and/or notifications.	HAR/SHAZAM	Set Alert and Notification Settings	AlertAndNotificationHelper:notifyAndAlert SD
SR1.5.2.1.9	The system shall support configuration parameters for SHAZAM devices.	HAR/SHAZAM	Set SHAZAM Configuration	GUISHAZAMClases CD
SR1.5.2.1.9.11	Specify SHAZAM Configuration	HAR/SHAZAM	Set SHAZAM Configuration	GUISHAZAMClases CD
SR1.5.2.1.9.11.6	[CHANGED AND MOVED TO 1.5.2.1.9.11.12.2] The system shall allow the user to specify the default phone number of the SHAZAM (10, 11, or 12 digits).	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.9.11.7	[CHANGED AND MOVED TO 1.5.2.1.9.11.12.1.4] The system shall allow the user to specify the access code for the SHAZAM.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.9.11.9	The system shall allow the user to specify the polling (refresh) interval for a SHAZAM.	HAR/SHAZAM	Set SHAZAM Configuration	SHAZAMReqHdlr:processEditCommsSettings SD SHAZAMControlModule.setConfiguration SD
SR1.5.2.1.9.11.9.4	The default polling (refresh) interval when adding a HWG-ER02a SHAZAM shall be 5 minutes. (The user can override this value)	HAR/SHAZAM	Set SHAZAM Configuration	Use Case Only
SR1.5.2.1.9.11.10	[CHANGED AND MOVED TO 1.5.2.1.9.11.12.1.3] The system shall allow the user to specify the port manager configuration for the SHAZAM, as defined by the Specify Port Manager Configuration requirements.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.9.11.11	[CHANGED AND MOVED TO 1.5.2.1.9.11.12.1.1] The system shall indicate to the user that a Telephony-type port will be used when communicating with the SHAZAM.	HAR/SHAZAM	N/A (Moved)	N/A (Moved)
SR1.5.2.1.9.11.12	The system shall allow the user to specify the model of the SHAZAM.	HAR/SHAZAM	Set SHAZAM Configuration	SHAZAMReqHdlr:processAddSHAZAM SD SHAZAMReqHdlr:processChangeSHAZAMModel SD SHAZAMControlModule.createSHAZAM SD SHAZAMControlModule.changeModelType SD
SR1.5.2.1.9.11.12.1	The system shall support the Viking RC2A SHAZAM model.	HAR/SHAZAM	Set SHAZAM Configuration	SHAZAMReqHdlr:processAddSHAZAM SD SHAZAMReqHdlr:processChangeSHAZAMModel SD SHAZAMControlModule.createSHAZAM SD SHAZAMControlModule.changeModelType SD
SR1.5.2.1.9.11.12.1.1	The system shall indicate to the user that a Telephony-type port will be used when communicating with a Viking RC2A SHAZAM.	HAR/SHAZAM	Set SHAZAM Configuration Settings	SHAZAMReqHdlr:processEditCommsSettings SD
SR1.5.2.1.9.11.12.1.2	The system shall allow the user to specify the default phone number of the Viking RC2A SHAZAM (10, 11, or 12 digits).	HAR/SHAZAM	Set Telephony Communication Settings	SHAZAMReqHdlr:processEditCommsSettings SD
SR1.5.2.1.9.11.12.1.3	The system shall allow the user to specify the port manager configuration for the Viking RC2A SHAZAM, as defined by the Specify Port Manager Configuration requirements.	HAR/SHAZAM	Set Telephony Communication Settings	Use Case Only
SR1.5.2.1.9.11.12.1.4	The system shall allow the user to specify the access code for the Viking RC2A SHAZAM.	HAR/SHAZAM	Set SHAZAM Configuration	SHAZAMReqHdlr:processEditCommsSettings SD
SR1.5.2.1.9.11.12.2	The system shall support the HWG-ER02a SHAZAM model.	HAR/SHAZAM	Set SHAZAM Configuration	SHAZAMReqHdlr:processAddSHAZAM SD SHAZAMReqHdlr:processChangeSHAZAMModel SD SHAZAMControlModule.createSHAZAM SD SHAZAMControlModule.changeModelType SD
SR1.5.2.1.9.11.12.2.1	The system shall indicate to the user that a TCP/IP port will be used when communicating with the HWG-ER02a SHAZAM.	HAR/SHAZAM	Set SHAZAM Configuration Settings	SHAZAMReqHdlr:processEditCommsConfig SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.9.11.12.2.2	The system shall allow the user to specify the IP address and port used to communicate to the HWG-ER02a SHAZAM.	HAR/SHAZAM	Set TCPIP Communication Settings	SHAZAMReqHdlr:processEditCommsConfig SD
SR1.5.2.1.9.11.12.2.5	The system shall allow the user to specify the relay that is to be used on the HWG-ER02a SHAZAM. (The device has two relays, only one of which will be used).	HAR/SHAZAM	Configure HWG-ER02a SHAZAM	GUIShazamClasses CD
SR1.5.2.1.9.11.12.3	The system shall require that a SHAZAM be offline before the model can be changed for a SHAZAM that already exists in the system.	HAR/SHAZAM	Change SHAZAM Model	SHAZAMReqHdlr:processChangeSHAZAMModel SD SHAZAMPushConsumer:handleModelChanged SD SHAZAMControlModule.changeModelType SD
SR1.5.2.1.9.11.13	The system shall allow a suitably privileged user to specify alert and notification settings for a SHAZAM.	HAR/SHAZAM	Set SHAZAM Configuration	SHAZAMReqHdlr:processSetAlertAndNotification Settings SD
SR1.5.2.1.9.11.13.1	The system shall support setting a responsible Center for a SHAZAM which is to receive the Device Failure Alert when the device goes into hardware failure.	HAR/SHAZAM	Set Alert and Notification Settings	SHAZAMReqHdlr:processSetAlertAndNotification Settings SD
SR1.5.2.1.9.11.13.2	The system shall support setting a responsible Center for a SHAZAM which is to receive the Device Failure Alert when the device goes into communications failure.	HAR/SHAZAM	Set Alert and Notification Settings	SHAZAMReqHdlr:processSetAlertAndNotification Settings SD
SR1.5.2.1.9.11.13.3	The system shall support setting a notification group to receive SHAZAM communication failure notification messages.	HAR/SHAZAM	Set Alert and Notification Settings	SHAZAMReqHdlr:processSetAlertAndNotification Settings SD
SR1.5.2.1.9.11.13.4	The system shall support setting a notification group to receive SHAZAM hardware failure notification messages.	HAR/SHAZAM	Set Alert and Notification Settings	SHAZAMReqHdlr:processSetAlertAndNotification Settings SD
SR1.5.2.1.9.11.13.5	The system shall support setting an operations center or notification group to "None" to disable the sending of alerts and/or notifications.	HAR/SHAZAM	Set Alert and Notification Settings	SHAZAMReqHdlr:processSetAlertAndNotification Settings SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.9.12	Add / Copy SHAZAM	HAR/SHAZAM	Add SHAZAM	N/A (Heading)
SR1.5.2.1.9.12.5	The system shall allow a suitably privileged user to add an HWG-ER02a SHAZAM.	HAR/SHAZAM	Add SHAZAM	SHAZAMReqHdlr:processAddSHAZAM SD SHAZAMControlModule.createSHAZAM SD
SR1.5.2.10	The system shall allow the user with appropriate rights to put equipment on-line in CHART.		N/A	N/A (Existing)
SR1.5.2.10.2	The system shall allow a suitably privileged user to set a HAR online.	HAR/SHAZAM	Put HAR Online	N/A (Existing)
SR1.5.2.10.2.14	The system shall allow a user with the Maintain HAR or Manage Response Plan user rights to initiate a poll of a HAR device if the device is online and the HAR model and communications type supports polling.	HAR/SHAZAM	Poll DR1500 HAR	DR1500HARReqHdlr:processPollHARNow SD
SR1.5.3	PERFORM ROUTINE MAINTENANCE. The system shall allow the user with appropriate rights to view the device status, and know why it's not on-line (including the key trouble ticket information) and know the problem is being addressed. The system shall also allow the user to take the device offline of maintenance or other adjustments including resetting the controller. Suggestion/example to be validated: e.g., integrate device maintenance web pages with CHART.		N/A	N/A (Heading)
SR1.5.3.1	The system shall allow a suitably privileged user to place a device in maintenance mode.		N/A	N/A (Existing)
SR1.5.3.1.2	The system shall allow a suitably privileged user to set a HAR to maintenance mode.	HAR/SHAZAM	Put HAR In Maintenance Mode	Use Case Only

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.3.1.2.12	The system shall allow a user with the Maintain HAR user right to initiate a poll of a HAR device if the device is in maintenance mode and the HAR model and communications type supports polling.	HAR/SHAZAM	Poll DR1500 HAR	SystemInterfaces/HARControl.cad HARControlModule:pollHARS DR1500HARReqHdlr:processPollHAR DR1500HARReqHdlr:processEditDR1500HARCtrlSettings HARContorlModule:ProcessPollResults SD HISDR1500ProtocolHdlr:getHARVersionInformation SD HISDR1500ProtocolHdlr:getHARModeAndSubMode SD HISDR1500ProtocolHdlr:getTransmitterStatus SD HISDR1500ProtocolHdlr:getTransmitterMode SD HISDR1500ProtocolHdlr:getSystemStatus SD HISDR1500ProtocolHdlr:parseLastCommandTimeStampFromResponse SD HISDR1500ProtocolHdlr:getLastCmdTimeStampFromResponse SD
SR1.5.3.8	The system shall allow a suitably privileged user to play a message from a selected slot in the HAR controller.	HAR/SHAZAM	View HAR Slot Usage	Use Case Only
SR1.5.3.8.2	The system shall allow a suitably privileged user to monitor (listen to) the contents of a specified slot of an independent HIS DR1500 HAR, if the HAR is not offline, and the port type is telephony.	HAR/SHAZAM	View HAR Slot Usage	Use Case Only
SR1.5.3.8.3	The system shall allow a suitably privileged user to monitor (listen to) the contents of a specified slot of an individual constituent HAR of a Synchronized HAR, if the constituent HAR is not offline and the constituent HAR supports monitoring.	HAR/SHAZAM	View HAR Slot Usage	Use Case Only
SR1.5.4	RESPOND TO EQUIPMENT/ DEVICE OUTAGE. The system shall allow the user with appropriate rights to notify maintenance personnel of an equipment outage that they have detected (or has been brought to their attention).		N/A	N/A (Heading)

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.4.1	The system shall inform the operators of a HAR failure.	HAR/SHAZAM	N/A	SystemInterfaces/HARControl.cad HARControlModule:pollHARS DR1500HARReqHdlr:processPollHAR DR1500HARReqHdlr:processEditDR1500HARCtrlSettings HARContorlModule:ProcessPollResults SD HISDR1500ProtocolHdlr:getHARVersionInformation SD HISDR1500ProtocolHdlr:getHARModeAndSubMode SD HISDR1500ProtocolHdlr:getTransmitterStatus SD HISDR1500ProtocolHdlr:getTransmitterMode SD HISDR1500ProtocolHdlr:getSystemStatus SD HISDR1500ProtocolHdlr:parseLastCommandTimeStampFromResponse SD HISDR1500ProtocolHdlr:getLastCmdTimeStampFromResponse SD
SR1.5.4.1.2	The system shall inform operators of a HAR hardware failure for a HIS DR1500 HAR that uses TCP/IP communications.	HAR/SHAZAM	Poll DR1500 HAR	SystemInterfaces/HARControl.cad HARControlModule:pollHARS DR1500HARReqHdlr:processPollHAR DR1500HARReqHdlr:processEditDR1500HARCtrlSettings
SR1.5.4.1.2.1	The system shall consider a HIS DR1500 HAR hardware failed if its DC voltage is below the minimum DC voltage threshold as configured in CHART for the HAR.	HAR/SHAZAM	Poll DR1500 HAR Set DR1500 Hardware Failure Detection Settings	SystemInterfaces/HARControlDR1500.cad HISDR1500ProtocolHdlr:getStatus SD HISDR1500ProtocolHdlr:getSystemStatus SD
SR1.5.4.1.2.4	The system shall consider a HIS DR1500 HAR hardware failed if its voltage standing wave ratio (VSWR) is above the maximum threshold as configured in CHART for the HAR.	HAR/SHAZAM	Poll DR1500 HAR Set DR1500 Hardware Failure Detection Settings	SystemInterfaces/HARControlDR1500.cad HISDR1500ProtocolHdlr:getStatus SD HISDR1500ProtocolHdlr:getTransmitterStatus SD
SR1.5.4.1.2.5	The system shall consider a HIS DR1500 HAR hardware failed if its status indicates that the power to the HAR is off.	HAR/SHAZAM	Poll DR1500 HAR	SystemInterfaces/HARControlDR1500.cad HISDR1500ProtocolHdlr:getStatus SD HISDR1500ProtocolHdlr:getSystemStatus SD
SR1.5.4.7	The system shall generate a Device Failure Alert for all DMSs, TSSs, HARs, and SHAZAMs capable of reporting that they are experiencing a hardware failure	HAR/SHAZAM	N/A	SystemInterfaces/AlertManagement.cad

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.4.8	The system shall inform the operators of a SHAZAM device failure.	HAR/SHAZAM	N/A	N/A (General)
SR1.5.4.8.1	The system shall inform the operators of a SHAZAM communications failure.	HAR/SHAZAM	Reset SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD AlertAndNotificationHelper.notifyAndAlert SD
SR1.5.4.8.2	The system shall inform operators of a SHAZAM hardware failure for a HWG-ER02a SHAZAM.	HAR/SHAZAM	Reset HWGER02a SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD AlertAndNotificationHelper.notifyAndAlert SD
SR1.5.4.8.2.1	The system shall consider a HWG-ER02a SHAZAM hardware failed if after setting the state of its relay the device does not indicate the relay is in the commanded state. (For example, if the system commands the device to close Relay 1 and then queries Relay 1 and it is open, the system shall consider the device failed.)	HAR/SHAZAM	Reset HWGER02a SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.setBeaconsState SD AlertAndNotificationHelper.notifyAndAlert SD
SR1.5.4.9	The system shall support automatic generation of a Device Failure Notification for all DMS, TSS, HAR, and SHAZAM devices that support being polled for status.	HAR/SHAZAM	N/A	N/A (General)
SR1.5.4.9.1	The system shall send a notification to the communication failure notification group configured for the device (if any) when the device transitions to a communication failed status.	HAR/SHAZAM	Poll DR1500 HAR Reset SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD AlertAndNotificationHelper.notifyAndAlert SD
SR1.5.4.9.2	The system shall send a notification to the communication failure notification group configured for the device (if any) when a HAR or SHAZAM device transitions from a communication failed status to a status of OK.	HAR/SHAZAM	Poll DR1500 HAR Reset SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.handleOpStatus SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.4.9.3	The system shall send a notification to the communication failure notification group configured for the device (if any) when a HAR or SHAZAM device transitions from a communication failed status to a status of hardware failed if the communication failure notification group configured for the device is different than the hardware failure notification group configured for the device. (This prevents duplicate notifications; if the groups are different, that means when the device transitions to hardware failed it is no longer the communication failure group's responsibility. If the groups are the same, the group will receive the hardware failure notification due to requirement SR1.5.4.9.4.)	HAR/SHAZAM	Poll DR1500 HAR Reset SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.handleOpStatus SD AlertAndNotificationHelper.notifyAndAlert SD
SR1.5.4.9.4	The system shall send a notification to the hardware failure notification group configured for the device (if any) when the device transitions to a hardware failed status.	HAR/SHAZAM	Poll DR1500 HAR Reset SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.handleOpStatus SD AlertAndNotificationHelper.notifyAndAlert SD
SR1.5.4.9.5	The system shall send a notification to the hardware failure notification group configured for the device (if any) when a HAR or SHAZAM device transitions from a hardware failed status to a status of OK.	HAR/SHAZAM	Poll DR1500 HAR Reset SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.handleOpStatus SD AlertAndNotificationHelper.notifyAndAlert SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.4.9.6	The system shall send a notification to the hardware failure notification group configured for the device (if any) when a HAR or SHAZAM device transitions from a hardware failed status to a status of communication failed if the hardware failure notification group configured for the device is different than the communication failure notification group configured for the device. (This prevents duplicate notifications; if the groups are different, then when the device transitions to communication failed it is no longer the recipient's responsibility. If the groups are the same, the recipients will receive the communication failure notification due to preceding requirements.)	HAR/SHAZAM	Poll DR1500 HAR Reset SHAZAM to Last Known State	SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.handleOpStatus SD SHAZAMControlModule.setBeaconsState SD AlertAndNotificationHelper.notifyAndAlert SD
SR1.5.5	VIEW DEVICE LISTS		N/A	N/A (Heading)
SR1.5.5.2	The system shall allow the user to view the list of HARs that exist in the system.	HAR/SHAZAM	View HAR List	Use Case Only
SR1.5.5.2.1	The system shall allow the user to view detailed data for each HAR in the list.	HAR/SHAZAM	View HAR List	Use Case Only
SR1.5.5.2.1.10	The detailed data displayed for a HAR shall include the Port Managers assigned to a HAR, if the HAR is configured to use a Telephony port for communications.	HAR/SHAZAM	View HAR List	Use Case Only
SR1.5.5.3	The system shall allow the user to view the list of SHAZAMs that exist in the system.	HAR/SHAZAM	View SHAZAM List	Use Case Only
SR1.5.5.3.1	The system shall allow the user to view detailed data for each SHAZAM in the list	HAR/SHAZAM	View SHAZAM List	Use Case Only
SR1.5.5.3.1.10	The detailed data displayed for a SHAZAM shall include the Port Managers assigned to a SHAZAM if the SHAZAM is configured to use a	HAR/SHAZAM	View SHAZAM List	Use Case Only

Tag	Requirement	Feature	Use Cases	Other Design Elements
	Telephony port for communications.			
SR1.5.7	VIEW DEVICE DETAILS		N/A	Use Case Only
SR1.5.7.2	View HAR Details	HAR/SHAZAM	N/A	Use Case Only
SR1.5.7.2.1	The system shall allow a suitably privileged user to view the configuration and status information of a HAR in the system.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.4	The system shall show the operational status of the HAR.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.4.1	Operational status shall indicate whether the HAR is online, in maintenance mode, or offline.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.4.2	Operational status shall indicate if the HAR has a communications failure.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.4.3	Operational status shall indicate whether the HAR is hardware failed.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.9	The system shall show the transmitter state of the HAR. (On/Off)	HAR/SHAZAM	View HAR Details	SystemInterfaces/HARControlDR1500.cad HISDR1500ProtocolHdlr:getTransmitterStatus SD
SR1.5.7.2.1.16	The system shall display the phone number for the HAR controller if the HAR is configured to use a Telephony port for communications.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.18	The system shall display the IP address and port if the HAR is configured to use TCP/IP for communications.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.20	The system shall display the hardware failure thresholds for a HIS DR1500 HAR when it is set to use TCP/IP communications.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.20.1	The system shall display the minimum voltage threshold for a HIS DR1500 HAR set to use TCP/IP communications.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.20.4	The system shall display the maximum voltage standing wave ratio (VSWR) threshold for a HIS DR1500 HAR that is set to use TCP/IP communications.	HAR/SHAZAM	View HAR Details	Use Case Only

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.7.2.1.21	The system shall display model specific status information for a HIS DR1500 HAR that uses TCP/IP communications.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.21.1	The system shall display a timestamp for the last time a status mismatch was detected (if any) for a HIS DR1500 HAR. (A status mismatch is when the system queries the status of the HAR and finds it does not match the status as commanded by the system.)	HAR/SHAZAM	View HAR Details	HISDR1500ProtocolHdlr:getLastCmdTimeStamp SD AP55AndDR1500HARCommand:parseLastCommandTimeStampFromResponse SD HARProtocolsPkg/HARProtocolsPkg.cad
SR1.5.7.2.1.21.2	The system shall display the power status (on/off) for a HIS DR1500 HAR.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.21.3	The system shall display the DC voltage of a HIS DR1500 HAR.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.21.4	The system shall display the broadcast monitor percent of full scale for a HIS DR1500 HAR.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.21.5	The system shall display the mode of a HIS DR1500 HAR. (Values can include Off, Play List, Alert, Live, Aux, and Other)	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.21.6	The system shall display the sub-mode of a HIS DR1500 HAR. (e.g. Synchronized, Unsynchronized)	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.21.7	The system shall display the transmitter status of a HIS DR1500 HAR. (Set Power, Forward Power, Reflected Power, Voltage Standing Wave Ratio (VSWR), and Modulation)	HAR/SHAZAM	View HAR Details	SystemInterfaces/HARControlDR1500.cad HARProtocolsPkg/HARProtocolsPkg.cad HISDR150ProtocolHdlr:getTransmitterStatus SD
SR1.5.7.2.1.21.8	The system shall display the version information for the digital communications console (DCC) of a HIS DR1500 HAR.	HAR/SHAZAM	View HAR Details	HISDR1500ProtocolHdlr:getHARVersionInformation SD
SR1.5.7.2.1.21.9	The system shall display the version information for the HAR module of a HIS DR1500 HAR. (Software version, build, EPROM checksum.)	HAR/SHAZAM	View HAR Details	HISDR1500ProtocolHdlr:getHARVersionInformation SD
SR1.5.7.2.1.21.	The system shall display the timestamp	HAR/SH	View HAR Details	AP55AndDR1500HARCommand:parseLastComma

Tag	Requirement	Feature	Use Cases	Other Design Elements
10	from a HIS DR1500 HAR. (This is the timestamp of the last change to the HAR state as reported by the HAR.)	AZAM		ndTimeStampFromResponse SD HISDR1500ProtocolHdlr:getLastCmdTimeStamp
SR1.5.7.2.1.21.11	The system shall display the sync mode from the HIS DR1500 HAR. (The sync mode will be ON, OFF, or N/A, where N/A is used if the HAR sub-mode is not synchronized).	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.22	The system shall display an indicator to show if automatic polling is enabled for the HAR if the HAR is set to use TCP/IP for communications.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.22.1	The system shall display the polling interval if polling is enabled.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.2.1.23	The system shall display the last time the Setup HAR command was successfully executed by the system.	HAR/SHAZAM	View HAR Details	Use Case Only
SR1.5.7.6	View SHAZAM Details		N/A	N/A (Heading)
SR1.5.7.6.2	The system shall allow a suitably privileged user to view the configuration and status information of a SHAZAM in the system.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.1	The system shall display the name of the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.2	The system shall display the model of the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.3	The system shall display the static message displayed on the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.4	The system shall display the owning organization of the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.5	The system shall display the maintaining organization of the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.6	The system shall display the network connection site of the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.7	The system shall display the location of the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8	The system shall display the communications settings for the	HAR/SHAZAM	View SHAZAM Details	Use Case Only

Tag	Requirement	Feature	Use Cases	Other Design Elements
	SHAZAM.			
SR1.5.7.6.2.8.1	The system shall display the communications port type used to communicate with the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8.2	The system shall display the Telephony related communications settings when the SHAZAM is configured to use a Telephony port for communications.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8.2.1	The system shall display the default phone number for the SHAZAM when the SHAZAM is configured to use a Telephony port for communications.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8.2.2	The system shall display the access code for the SHAZAM when the SHAZAM is configured to use a Telephony port for communications.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8.2.3	The system shall display the port manager timeout for the SHAZAM when the SHAZAM is configured to use a Telephony port for communications.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8.2.4	The system shall display the port managers and associated phone numbers for the SHAZAM when the SHAZAM is configured to use a Telephony port for communications.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8.3	The system shall display the TCP/IP related communications settings when the SHAZAM is configured to use TCP/IP communications.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.8.3.1	The system shall display the IP Address and Port for the SHAZAM when the SHAZAM is configured to use TCP/IP communications.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.9	The system shall display the auto-refresh settings for the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.9.1	The system shall indicate if auto-refresh is enabled for the SHAZAM.	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR1.5.7.6.2.9.2	The system shall display the auto-refresh	HAR/SH	View SHAZAM Details	Use Case Only

Tag	Requirement	Feature	Use Cases	Other Design Elements
	rate if auto-refresh is enabled for the SHAZAM.	AZAM		
SR1.5.7.6.2.10	The system shall display the relay being used on a HWG-ER02a SHAZAM. (Relay 1 or Relay 2)	HAR/SHAZAM	View SHAZAM Details	Use Case Only
SR7	MANAGE CHART PERFORMANCE. This process allows CHART managers and others to assess and enhance the effectiveness of CHART by reviewing and evaluating the performance of CHART operations, event management, traffic flow management, and devices and software performance. This process also includes simulation of event management and traffic management based on historical data.		N/A	N/A (Heading)
SR7.3	MANAGE AND MEASURE DEVICE PERFORMANCE		N/A	N/A (Heading)
SR7.3.2	CHECK AND VALIDATE SYSTEM AND STATUS. The system shall initiate the capture of data from polling for devices and hardware.		N/A	N/A (Heading)
SR7.3.2.1	The system shall monitor and ping for system services at pre-defined periodicity.		N/A	N/A (General)
SR7.3.2.1.1	The system shall provide a mechanism to poll devices for status on a configurable interval basis.		N/A	N/A (General)
SR7.3.2.1.1.4	The system shall support polling of HAR devices.	HAR/SHAZAM	Poll DR1500 HAR	SystemInterfaces/HARControl.cad HARControlModule:pollHARS DR1500HARReqHdlr:processPollHAR DR1500HARReqHdlr:processEditDR1500HARCtrlSettings
SR7.3.2.1.1.4.1	Polling of HAR devices shall be supported for HARs configured to use TCP/IP communications.	HAR/SHAZAM	Poll DR1500 HAR	SystemInterfaces/HARControl.cad HARControlModule:pollHARS DR1500HARReqHdlr:processPollHAR DR1500HARReqHdlr:processEditDR1500HARCtrlSettings

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR7.3.2.1.1.4.2	The system shall verify that the message currently being broadcast by a HAR is the message the system expects to be broadcast on the HAR for HAR devices supporting the appropriate level of status information.	HAR/SHAZAM	Maintain HAR State	DR1500HARReqHdlr:processPollHAR
SR7.3.2.1.1.5	The system shall support the polling of SHAZAM devices.	HAR/SHAZAM	Reset SHAZAM to Last Known State Reset HWG ER02a SHAZAM to Last Known State	SHAZAMControlModule.RefreshSHAZAMInBackground SD SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.setBeaconsState SD
SR7.3.2.1.1.5.1	Polling of SHAZAM devices shall be supported for SHAZAM models that provide a means to query the device status.	HAR/SHAZAM	Reset SHAZAM to Last Known State Reset HWG ER02a SHAZAM to Last Known State	SHAZAMControlModule.RefreshSHAZAMInBackground SD SHAZAMControlModule.refreshImpl SD SHAZAMControlModule.setBeaconsState SD
SR9	SYSTEM MAINTAINABILITY, AVAILABILITY, SECURITY, AND DATA DISTRIBUTION		N/A	N/A (Heading)
SR9.4	Data Distribution		N/A	N/A (Heading)
SR9.4.2	The system shall protect certain information maintained in the system by limiting a user's access to /view of that information based on the user's functional rights.		N/A	N/A (General)
SR9.4.2.8	The system shall allow a suitably privileged user to view sensitive configuration data associated with a HAR (View HAR Sensitive Config functional right).	HAR/SHAZAM	View HAR Details	Use Case Only
SR9.4.2.8.1	Sensitive configuration data for a HAR includes: Default Phone Number, Access Code, Port Manager Connection Timeout, Port Type, per Port Manager phone numbers, IP Address and Port.	HAR/SHAZAM	View HAR Details	Use Case Only
SR9.4.2.9	The system shall allow a suitably privileged user to view sensitive configuration data associated with a	HAR/SHAZAM	View SHAZAM Details	Use Case Only

Tag	Requirement	Feature	Use Cases	Other Design Elements
	SHAZAM (View HAR Sensitive Config functional right.)			
SR9.4.2.9.1	Sensitive configuration data for a SHAZAM includes: Default Phone Number, Access Code, Port Manager Connection Timeout, Port Type, per Port Manager phone numbers, IP Address and Port.	HAR/SHAZAM	View SHAZAM Details	Use Case Only

9Acronyms/Glossary

Alert	A feature of the CHART system used to alert operators of important information or events via a panel in their GUI and audible messages.
Broadcast Monitor Percent	A measurement provided by the DR1500 HAR that indicates the output level from the recorder component to the radio component. A typical value is 80%.
DCC	Digital Communication Controller, a module that can be added to a DR1500 HAR to allow it to be controlled via TCP/IP
Firmware	The fixed program
GUI	Graphical User Interface; the part of the CHART system accessed via a web browser to provide users access to the system.
HAR	Highway Advisory Radio
IP	Internet Protocol, the method (or protocol) by which data is sent from one computer to another on the Internet.
Modulation	Amplitude Modulation (AM) is a technique used in electronic communication that works by varying the strength of the transmitter signal in relation to the information being sent.
Notification	A feature of the CHART system used to notify groups or individuals of important information or events. This is implemented via a third party notification engine named "!Attention" and the notifications are typically sent to end users as e-mail.
Relay	An electrically operated switch. In the context of a SHAZAM, it is used to switch the beacons on or off.
SHAZAM	A highway sign that contains beacons that can be activated to indicate that a message is active on a HAR. Typically the face of the sign contains text such as "Tune Radio To …"
Synchronized HAR	A HAR that is actually a collection of two or more HARs, known as constituent HARs, that are operated as if they are a single HAR. A message set on a Synchronized HAR is set on each of its constituents, and the constituents are set in a mode that causes them to synchronize their broadcast so they are broadcasting the message at exactly the same time.
TCP	Transmission Control Protocol, a protocol used as a layer on top of IP to send data as message units from one computer to another on the Internet.
VSWR	Voltage Standing Wave Ratio, a measurement from a radio transmitter of the impedance mismatch between the transmission line and its load. The higher the VSWR, the greater the mismatch.