JTLS-GO

Version Description Document

June 2026

DEPARTMENT OF DEFENSE
JOINT STAFF J7
116 LAKE VIEW PARKWAY
SUFFOLK, VA 23435-2697

JOINT THEATER LEVEL SIMULATION - GLOBAL OPERATIONS
(JTLS-GO 6.3.10.0)

[Blank Page]

ABSTRACT

The Joint Theater Level Simulation - Global Operations (JTLS-GO®) is an interactive, computer-based, multi-sided wargaming system that models air, land, naval, Special Forces, and Non-Governmental Organization (NGO) functions within a combine joint and coalition environment.

This *JTLS-GO Version Description Document (VDD)* describes the new features of the Version 6.3.10.0 delivery of the configuration-managed JTLS-GO software suite.

JTLS-GO 6.3.10.0 is a Maintenance release of the JTLS-GO 6.3 series that includes fixes to uncovered software issues, an updated repository of standard data, and a demonstration scenario based in the western Pacific. This release includes one new model enhancement, described in Chapter 2. Code modifications that represent corrections to known Software Trouble Reports (STRs) are described in Chapter 3. Remaining and outstanding STRs are described in Chapter 4.

This publication is updated and revised as required for each Major or Maintenance version release of the JTLS-GO model. Corrections, additions, or recommendations for improvement must reference specific sections, pages, and paragraphs with appropriate justification and be forwarded to:

JTLS-GO Director of Development
Valkyrie Enterprises LLC
1200 Piedmont Ave, Suite A
Pacific Grove, California 93950-5157
United States
jtlsgo@valkyrie.com

[Blank Page]

TABLE of CONTENTS

1.0 INTRODUCTION

1.1 SCOPE

This *JTLS-GO Version Description Document* (VDD) describes Version 6.3.10.0 of the configuration managed Joint Theater Level Simulation - Global Operations (JTLS-GO®) software suite. JTLS-GO 6.3.10.0 is a Maintenance delivery for the JTLS-GO 6.3 series of releases.

JTLS-GO 6.3.10.0 includes the entire JTLS-GO suite of software, a repository of engineering level data, and a realistic demonstration scenario, called “wespac63”, based on the Western Pacific theater of operations. There were no database format modifications between this Maintenance release and the original JTLS-GO 6.3.0.0 version. Appendix B of the original *JTLS-GO 6.3.0.0 Version Description Document* summarized the database format changes made between the JTLS-GO 6.2 series and this JTLS-GO 6.3 series of the software system.

Detailed description of the new Engineering Change Proposals (ECPs) is provided in Chapter 2.0. Chapter 3.0 summarizes the Software Trouble Reports (STRs) that have been corrected and are delivered with this version of JTLS-GO 6.3.

JTLS-GO 6.3.10.0 executes on the Red Hat Enterprise Linux Version 9.4 and Oracle Linux 9.4 64-bit operating systems. The Web-Hosted Interface Program (WHIP®) user workstation interface can be executed on any 64-bit operating system from any Java-compatible Web browser.

1.2 INVENTORY OF MATERIALS

This section lists documents and software that are relevant to JTLS-GO. All JTLS-GO documents included in this delivery are provided in PDF format within a documents subdirectory.

1.2.1 Obsolete/Outdated Documents

No documents have been deleted or become outdated as a result of this release.

1.2.2 Unchanged Documents

- *JTLS-GO Analyst Guide* (JTLS-GO Document 01, Version 6.3.8.0)
- *JTLS-GO Air Services User Guide (*JTLS-GO Document 02, Version 6.3.4.0)
- *JTLS-GO Configuration Management Plan* (JTLS-GO Document 03, Version 6.3.0.0)
- *JTLS-GO Controller Guide* (JTLS-GO Document 04, Version 6.3.8.0)
- *JTLS-GO Data Requirements Manual* (JTLS-GO Document 05, Version 6.3.8.0)
- *JTLS-GO DDS User Guide* (JTLS-GO Document 06, Version 6.3.6.0)

- *JTLS-GO Director Guide* (JTLS-GO Document 07, Version 6.3.7.0)
- *JTLS-GO Executive Overview* (JTLS-GO Document 08, Version 6.3.8.0)
- *JTLS-GO Installation Manual* (JTLS-GO Document 09, Version 6.3.9.0)
- *JTLS-GO WHIP Training Manual* (JTLS-GO Document 10, Version 6.3.8.0)
- *JTLS-GO Repository Description* (JTLS-GO Document 14, Version 6.3.4.0)
- *JTLS-GO Player Guide* (JTLS-GO Document 12, Version 6.3.9.0)
- *JTLS-GO Software Maintenance Manual* (JTLS-GO Document 15, Version 6.3.9.0)
- *JTLS-GO Entity Level Server User Guide* (JTLS-GO Document 19, Version 6.3.4.0)
- *JTLS-GO Federation User Guide* (JTLS-GO Document 20, Version 6.3.0.0)
- *JTLS-GO C4I Interface Manual* (JTLS-GO Document 21, Version 6.3.5.0)
- *JTLS-GO DoD Architecture Framework* (JTLS-GO Document 22, Version 6.3.0.0)

1.2.3 Updated Documents

- *JTLS-GO Technical Coordinator Guide* (JTLS-GO Document 16, Version 6.3.10.0)
- *JTLS-GO Version Description Document* (JTLS-GO Document 17, Version 6.3.10.0)

1.2.4 New Documents

No new documents are required for this version of the software.

1.2.5 Delivered Software Components

JTLS-GO 6.3.10.0 may be delivered either on a CD or as a set of compressed TAR files to be downloaded. Either method includes the complete suite of software executable code and command procedures. The following software components are included with this release:

- Combat Events Program (CEP)
- Geo-Spatial Service (GSS)
- Scenario Initialization Program (SIP)
- Interface Configuration Program (ICP)
- Reformat Spreadsheet Program (RSP)

- JTLS Symbols Application (JSYMS)
- Database Development System (DDS)

  Database Configuration Program (DCP)

  DDS Client User Interface (DDSC)
- ATO Translator Service (ATOT)
- ATO Generator Service (ATOG)
- ATO Retrieval Program (ATORET)
- JTLS Convert Location Program (JCONVERT)
- Count Critical Order Program (CCO)
- JTLS HLA Interface Program (JHIP)
- After Action Review Client (AARC)
- Scenario Data Client (SDC)
- Order Entry Client (OEC)
- Order Verification Tool (OVT)
- JTLS Object Distribution Authority (JODA)

  The current JODA build number is 215.
- Web Services Manager (WSM)
- Web-Hosted Interface Program (WHIP) and its component programs:

  Apache Server (APACHE) version 2.4.62

  JTLS XML Serial Repository (JXSR)

  Order Management Authority (OMA)

  Synchronized Authentication and Preferences Service (SYNAPSE)

  XML Message Service (XMS)

  Total Recall Interactive Playback Program (TRIPP)
- Entity Level Server (ELS)

- JTLS Operational Interface (JOI) for both OTH-Gold and Link 16 generation
- Tactical Electronic Intelligence (TACELINT) Message Service
- Keyhole Markup Language (KML) Operational Interface (KOI)
- JTLS Transaction Interface Program (JTOI)

  JTOI_ICC302 - Used to feed NATO Integrated Command Control (ICC) Version 3.0.2 system.

  JTOI_ICC320 - Used to feed NATO ICC Version 3.2.0 system.

  JTOI_ICC340 - Used to feed NATO ICC Version 3.2.0 system.

  JTOI_ICC350 - Used to feed NATO ICC Version 3.2.0 system.

  JTOI_NECCCIS - Used to feed NATO Northern European Command, Command Control Information System (NECCCIS).

  JTOI_TBMCS - Used to feed US Theater Battle Management Core System (TBMCS).

- JTLS Interface Network Navigator (JINN)
- JTLS Order of Battle Editor (JOBE)
- JTLS Geographic Information System (GIS) Terrain Building Program
- JTLS Master Integrated Database (MIDB) Tool
- JTLS Version Conversion Program (VCP)

  VCP60 - Converts a JTLS-GO 5.1 database to a JTLS-GO 6.0 formatted database.

  VCP61 - Converts a JTLS-GO 6.0 database to a JTLS-GO 6.1 formatted database.

  VCP62 - Converts a JTLS-GO 6.1 database to a JTLS-GO 6.2 formatted database.

  VCP63 - Converts a JTLS-GO 6.2 database to a JTLS-GO 6.3 formatted database.

Instructions for installing JTLS-GO 6.3.10.0 are provided in the *JTLS-GO Installation Manual*. Compared to the JTLS-GO 6.2 series, the JTLS-GO 6.3 series uses a significantly different version of PostgreSQL and the Linux operating system. If an organization has not already upgraded to the JTLS-GO 6.3 version, ensure special attention is given to following the documented operating system and PostgreSQL installation procedures. No other upgrade beyond installation of the compressed TAR files or CD is required. The software provided with this delivery is a complete release that includes all files and code required to execute JTLS-GO.

1.2.6 Released Databases

This release includes the following sample unclassified databases:

- The scenario that serves as a repository of engineering level data called “repository63”. Although not useful as a scenario, it does follow all of the database requirements for a scenario, and should be loaded into your PostgreSQL scenario table-space.
- The scenario “wespac63”, which is suitable for training and demonstrations. The scenario has been updated to use the newest version of engineering level data.

1.3 INTERFACE COMPATIBILITY

1.3.1 Support Software

JTLS-GO 6.3.10.0 requires the following versions of support software, including operating systems, compilers, scripting utilities, database tools, transfer protocols, and display managers.

- Operating system for the model: Red Hat Linux Enterprise Server (ES) Edition Version 9.4, 64-bit architecture.

  JTLS-GO 6.3 has been tested with the following versions of Linux 9:

  RedHat Linux 9.4 - this operating system license must be purchased.

  Oracle Linux 9.4 - This operating system is free to download, use, and distribute, and is provided in a variety of installation and deployment methods. It has been approved by Defense Information System Agency (DISA) for use by U.S. Government Agencies.

- There are no restrictions on the operating system for client workstations, except that the operating system must be a 64-bit architecture with a Java-enabled web browser. JTLS-GO 6.3.10.0 has been tested on the following operating systems:

  Red Hat Linux Enterprise Edition Version 9.4

  Oracle Linux 9.4

  Windows 10, which can be used only if the workstation is an external HTTP client of the simulation network.

- JTLS-GO 6.3.10.0 is delivered with the Adoptium project Temurin Java Development Kit (JDK) 1.8 Update 492 package. Both the ICP and DCP have the option for an organization to increase the maximum memory heap for the WHIP and DDSC. For large scenarios and databases, an organization should consider increasing the maximum heap size.
- JTLS-GO uses IcedTea to provide the Java Web Start capability that implements the web-enabled JTLS-GO functionality. JTLS-GO supports IcedTea version 1.8.4.

- JTLS-GO 6.3.10.0 is being delivered with PostgreSQL 15.16 that has been compiled under Linux 9.4 and is bundled with the tar files for this release. This version of PostgreSQL is the latest security patch release of PostgreSQL and is being delivered in accordance with US Department of Defense Cybersecurity requirements. It is not necessary to use the delivered solution, but it is the easiest method to meet the requirements of JTLS-GO 6.3.10.0. There are several alternative methods available for obtaining the PostgreSQL 15.16 software. Refer to Chapter 6 of the *JTLS-GO Installation Manual* for additional installation details.

- Windows software, X11R5 server, Motif 1.2 Library, Motif Window Manager: These items are included as part of the supported versions of Red Hat Linux ES.

- The Perl script language is used by the JTLS-GO system and game setup scripts. The version of Perl included with the supported versions of Red Hat Linux ES is sufficient. The Perl program is typically located in the /usr/bin directory. If Perl is installed in a another location, a link should be created from the /usr/bin directory to this program.

- SIMSCRIPT III (SIMSCRIPT to C) translator/compiler: SIMSCRIPT is required for recompiling JTLS-GO code. It is not necessary to have a SIMSCRIPT compiler to execute JTLS-GO, because all JTLS-GO software executables are statically linked with the SIMSCRIPT libraries. The compiler is needed only if you are a U.S. Government organization that can obtain source code and plan to re-compile JTLS-GO SIMSCRIPT code.

- ANSI C Compiler: It is not necessary to use a C compiler to execute JTLS-GO. This compiler is used only by U.S. Government organizations that can obtain source code and intend to re-compile any of the JTLS-GO component programs. The C Compiler version delivered with the supported versions of Red Hat Linux ES is sufficient.

- C++ Compiler: It is not necessary to use a C++ compiler to execute JTLS-GO. This compiler is used only by U.S. Government organizations that can obtain source code and intend to re-compile any of the JTLS-GO HLA component programs. The C++ Compiler version delivered with the supported versions of Red Hat Linux ES is sufficient.

- The JTLS-GO DDS application uses these open source libraries:

  JFreeChart, licensed under a GNU Lesser General Public License (LGPL) by Object Refinery Limited, http://www.object-refinery.com

  JCommon, licensed under LGPL2.1 (GNU Lesser General Public License version 2.1 or later) by Object Refinery Limited, http://www.object-refinery.com

  Commons-math3-3.0.jar, licensed under Apache Software Foundation (Apache License, Version 2.0) http://www.apache.org/licenses/LICENSE-2.0HLA Compliance

- KML Operational Interface (KOI)

The Keyhole Markup Language (KML) Operational Interface (KOI) server utility enables the model to feed operational simulation data to any version of Google Earth™. The display capabilities and data transfer features of this terrain viewer are sufficiently robust to be used as a base-level operational interface. Operational Players who may be restricted from using an operational Command, Control, Communication, Computer Information (C4I) systems may be able to install and use Google Earth and configure the KOI to provide a capability that resembles C4I for observing perception Force Side data.

Chapter 3 of the *JTLS-GO C4I Interface Manual* describes requirements and procedures for using the KOI capabilities.

- JTLS-GO 6.3.10.0, using the JODA service, allows connections and data exchange with customer client programs. The customer client programs are linked with a set of JTLS-GO-provided API libraries that permit a TCP/IP connection between the JODA and the client program. These API libraries, called JDSP libraries, are built for Linux and Windows and allow customers to built client applications on either of these operating systems. Below are the development environments under which each of the JDSP libraries are built:

  RedHat Linux 9.4 using gcc (GCC) 11.4.1 20231218 (Red Hat 11.4.1-3.0.1)

  Windows 10 using Visual Studio 2017 version 15.9.60 and Visual C++ 00369.60000.00001-AA807

1.3.2 JTLS-GO Cybersecurity Compliance

Because of recent incidents of intrusions into software systems, the United States Department of Defense (DoD) has implemented a strong and strictly enforced Cybersecurity program. JTLS-GO, as software that executes on DoD systems, must comply to the mandates of the program, along with all of the third party software used by JTLS-GO, such as PostgreSQL and Java.

One of the DoD requirements is that the software must implement a methodology that ensures that the end user keeps the software up-to-date and all security patches are properly installed. In previous versions of JTLS-GO, Java 8, as delivered by Oracle, fulfilled this mandate by implementing an expiration date for its software. The concept of an expiration date has been removed from the DoD requirement, but the concept of always using the latest version of third-party software remains a strong component of DoD Cybersecurity requirements.

The following procedure has been established and approved by the JS/J7 Cybersecurity branch to meet the software update requirement:

- Within days of an Oracle Java security release, AdoptOpenJDK produces an equivalent version using infrastructure, build and test scripts to produce pre-built binaries of the OpenJDK class libraries. All AdoptOpenJDK binaries and scripts are open source licensed and available for free.

- Within two-weeks of the AdoptOpenJDK release, JTLS-GO provides a bug release version (JTLS-GO 6.3.n.0) including a full Version Description Document (VDD) for download to all authorized agencies. All DoD agencies using JTLS-GO will be in full compliance with this specific Cybersecurity mandate as long as they download and use the bug released versions when distributed.

The JTLS-GO 6.3 series has been issued an Exit Gate letter and certification from the JS/J7 Cybersecurity branch. Please contact the U.S. Government Program Manager, Mr. Douglas Failor (douglas.l.failor.civ@mail.mil) to obtain the completed Cybersecurity paperwork.

1.3.3 JTLS-GO High Level Architecture Compliance

The JTLS-GO 6.3.10.0 release is fully High Level Architecture (HLA) compliant, and includes all the programs required to run JTLS-GO in an HLA mode. JTLS-GO currently belongs to one federation known as GlobalSim. GlobalSim is a comprehensive constructive simulation solution for joint training and wargaming that helps commanders and all levels of staff prepare for a range of operational scenarios.

The solution combines JTLS-GO with CAE's GESI constructive tactical entity-level simulation system. CAE's GESI constructive simulation system is designed to run complex and comprehensive exercises from the company level up to division level. The GESI system is used to represent a virtual battlefield, including weapons, vehicles, aircrafts, ground forces and more.

Combining JTLS-GO and GESI brings together operational and tactical level constructive simulations to prepare commanders and staff to make timely, informed and intelligent decisions across the full spectrum of operations, including conventional combat, disaster relief, and operations other than war.

From the JTLS-GO perspective, all software needed to run GlobalSim is included in this delivery. JTLS-GO uses the Federation Object Model (FOM) located in the $JGAME/data/hla directory. Previous federation testing of JTLS-GO with CAE's GESI model has been accomplished using this FOM. The reader should note that the JTLS-GO Development Team, to date, has not been able to test this federation using this maintenance release of JTLS-GO. If there is interest in running this federation, please contact the JTLS-GO Help desk at jtlsgo@valkyrie.com.

The HLA RTI (Run Time Infrastructure) executive program (rtiexec) recommended for use with this release is Pitch pRTI Evolved 4.4.2.0. However, this program is not included in the JTLS-GO 6.3.10.0 delivery. Users may obtain a full installation package of the RTI software from Pitch Corporation (www.pitch.se). For information about executing the HLA RTI Executive and other HLA-related software, refer to the appropriate HLA documentation and user guides.

1.4 DATABASE MODIFICATIONS

Significant database structure differences exist between the JTLS-GO 6.3 series and the previous JTLS-GO 6.2 series database structure. Appendix B of the *JTLS-GO 6.3.0.0 Version Description*

Document has a summary of all database changes. To upgrade your JTLS-GO 6.2 scenario to JTLS-GO 6.3 compatibility, see instructions listed in the *JTLS-GO DDS User Guide*, Chapter 3.1.

Due to changes made in support of STR JTLS-2025-17088 Missile Capability Does Not Have Torpedo, the record has been added to the JTLS-GO 6.3.5.0 Configuration Managed tw_missile_capability lookup table. This is not a database format change, but to make this new record available for the database building effort, users should **unload** their JTLS-GO 6.3 series scenarios prior to installing JTLS-GO 6.3.5.0, and then **load** them following installation. Please refer to the *JTLS-GO 6.3.5.0 Version Description Document* for more details.

Upgrading earlier JTLS-GO versions to the JTLS-GO 6.3 format will automatically correct the problem. No additional action is required.

Due to the mandated Cyber-Security improvements made to the Database Configuration Program (DCP) and the Database Development System Client (DDSC) for JTLS-GO 6.3.4.0, each PostgreSQL database scenario needs to be reconfigured. If you skipped moving in JTLS-GO 6.3.4.0, please execute the procedure in the following note for each JTLS-GO 6.3 scenario:

- Go to the $JTLSHOME/data/scenario/<scenario_name>/config directory.
- Delete the entire directory and its subdirectories.
- Start the new DCP and create the needed DDSCs.
- Each organization can decide whether they want to use the new encrypted password capability and the new DDSC privilege capability. Refer the *JTLS-GO Database Development User Guide* for information concerning the new DCP interface.

1.4.1 JTLS-GO Using Legacy Default Symbol Set

If a user organization is still using the pre-JTLS-GO 5.0.0.0 legacy default symbol set, prior to unloading your JTLS-GO 6.3.0.0 formatted data from your PostgreSQL database server into the JTLS-GO 6.3.0.0 scenario American Standard Code for Information Interchange (ASCII) text files, you must execute the JSYMS program using the procedure outlined in the *JTLS-GO DDS User Guide*, Appendix B.11. This procedure will reorganize the structure of the <scenario_name>.gs and databases symbol.scf file.

1.4.2 JTLS-GO Using New Default Symbol Set

Users are reminded that organizations should not make any modifications to the Default Symbol Set delivered with JTLS-GO. End-user organizations are free to use the Default Symbol Set in their

scenarios and alter the scenario symbol set to their scenarios to meet specific organizational needs.

The JTLS-GO Default Symbol Set has not been changed since August 2020. At that time, some new symbols were created to meet end-user requirements. No previously existing symbols were deleted nor were any of the preexisting symbol names changed. If any existing scenarios have not moved to this new Default Symbol Set, this means that the user can easily move in this new symbol set. Please follow the steps outlined in the *JTLS-GO DDS Users Guide*, Section B.13, Updating Scenario Symbol Set.

1.4.3 Standard Repository Changes

The JTLS-GO Database Team has continued to improve and expand the unclassified data repository, which has been renamed to “repository63”. The DDS comparison and synchronization function can be used to determine if any of the changes delivered are of use to a JTLS-GO user organization. Specifically, significant effort has been started to represent additional Combat Systems to more closely match the Combat Systems recognized by the Joint Live Virtual Constructive (JLVC) federation of models. This effort is expected to be an ongoing effort for the next three to five months.

1.5 INSTALLATION

The *JTLS-GO Installation Manual,* a Portable Document Format (pdf) file available for direct download, is part of this JTLS-GO delivery, It provides detailed instructions for installing the new version of JTLS-GO and the installation of PostgreSQL 15.13.

Reminder: Ensure existing JTLS-GO Version 6.3 databases held in the PostgreSQL table-space are unloaded prior to installation and reloaded after installation, as explained in Section 1.4.

2.0 ENGINEERING CHANGE PROPOSALS

No Engineering Change Proposals (ECP) were delivered with this release.

3.0 SOFTWARE TROUBLE REPORTS

Software Trouble Reports (STRs) describe software code errors that have been discovered by JTLS-GO users or developers and have been corrected.

3.1 JTLS-2026-17649 Restart ELS Specify Checkpoint Number

In the past, the ELS would restart from the same checkpoint number as was used by the CEP. This was not always the best checkpoint to use for the ELS if the CEP had been running for a long period of time.

The ELS code was modified to allow the user to specify which checkpoint should be used during a restart. The restart process remained the same, except that the checkpoint number was included as an argument when restarting the ELS. The javamenu and jtlsmenu were also updated to allow the user to specify which ELS checkpoints to restart from.

3.2 JTLS-2026-17650 Naval Move Crash

A Naval Move order containing a very long distance move was submitted to a Formation. The route traversed a large terrain grid that did not contain any objects of any kind. The model crashed when it attempted to determine if any barriers in the grid would cause a delay.

Because of the large number of terrain grids in a JTLS-GO scenario database, each terrain grid is assigned a set owner ONLY if there is at least one object in the grid. The set owner is assigned when the first object enters the grid. If there are no objects, then no set owner is established for that grid.

The routine that checks for barrier delays exited early because there was no set owner in the empty grid. The logic therefore bypassed the creation of a temporary great circle arc that would have been used to determine if the Formation crossed a barrier. At the exit point, the logic attempted to destroy the non-existent great circle arc, which caused the model to crash due a null pointer.

To correct this problem, a check was added to destroy the arc only if it exists.

3.3 JTLS-2026-17653 GIS Tool - Terrain Grid Misclassification

Several fictional islands were added to the database. When the user created a GIS Tool project, created terrain layers and grids, and then ran the tool's Calculate Grid Parameters, most of the grids on the islands were classified as Ocean when they clearly should have been classified as some other terrain type (Open, Forest, etc).

When the GIS Tool tried to process a polygon list object, it only processed one polygon in the list, not the entire list. This has been fixed so that the method processes every polygon in a polygon list.

3.4 JTLS-2026-17655 Airbase Gather Orphan Squadrons

An Airbase was magic moved to a location where a fixed-wing Squadron was located without an assigned home base. The Airbase and Squadron were from the same Force Side. Also, the Squadron was located within the Airbase radius after the magic move. The Squadron properly adopted the Airbase as its home base.

However, a code review revealed that there was no logic to Airbase the airbase if it happened to be from a different Force Side or if the Squadron was located outside the Airbase radius. Either of these situations could have caused complications during subsequent model processing.

The missing criteria was added to the routine to prevent a Squadron from adopting a home base from a different Force Side or located beyond the Airbase radius. This logic also applies during the unit arrival process if an Airbase arrives after the Squadron.

3.5 JTLS-2026-17656 UAVs Not Auto-Launched With Situation Change

If the situation changes for a Squadron, such as receiving more supplies, the model automatically determines if there are any air missions waiting for the supplies so they can launch. This check is currently only done for Squadrons, and not for other types of units that may own Unmanned Aerial Vehicles (UAVs).

The code only called the routine CAN YOU LAUNCH when supplies arrived at an Airbase, FARP, Naval Unit or Squadron. It did not call the routine for other types of units that could own aircraft. This was fixed.

The code was thoroughly reviewed to determine if there were any other situations in which non-Squadron units owning UAVs did not check whether waiting air missions could now launch because of a situation change. The following additional situations were found and fixed:

- If an Aircraft Class characteristic, such as usable fuel type, changes such that would make it not possible to launch the air mission.
- If the owning Unit was magic moved into an area that has needed supplies.
- If a portion of the Unit was placed on the ground as a result of an airlift or amphibious operation.
- If the Controller magically gave the Unit more available aircraft.
- If a Unit arrives in the game or is returned to the game, and there are already missions waiting to launch.
- If the weather is changed over a Unit with air missions waiting to launch.
- If one Unit attaches to another Unit, providing the surviving Unit more aircraft.

3.6 JTLS-2026-17659 Improve Navy Debug Output

When JTLS started development, the Simscript compiler did not have a debugging capability. To help with debugging, a Debug File was created for each type of object, Ground Unit, Naval Unit, Air Mission and Supply Run. Simscript has subsequently incorporated a debugger to help with debugging tasks.

The old Navy Debug file is not useful because it is easier to get the information through the improved Simscript debugging capability. What is needed is a Navy Debug file that tracks submarine detection calculations.

The old Navy Debug output was replaced with details concerning the computation of when a submarine will be detected. These are the steps needed to start the detection computation debug output:

1. From a Controller WHIP, turn on the Navy Debug Flag.

2. From a Controller WHIP, submit a Debug Object Order for the submarine whose detection time computation should be tracked.

3. Place the model in the debugger and set a breakpoint for SCHEDULE.COVERT.DETECTION at line 2. The model will indicate that there is no code at line 2 and will place the breakpoint at the first executable line. As of this writing that is line 53, but it may change if code changes to the routine are made.

4. Set game speed as desired.

5. When the model enters the debugger in SCHEDULE.COVERT.DETECTION, submit the following two commands:

   sv SAVED.DEBUG.FLAG #, where '#' is the side number of the detecting Force Side for which you are interested. For example, in the wespac64 scenario, Side 6 is the US side. If I wanted output for how Side 6 is computing its detection times for the ship named in Step 2, I would enter the command: **sv SAVED.DEBUG.FLAG 6**

   db 1, which will remove the breakpoint, so the model no longer enters the debugger each time it enters the routine SCHEDULE COVERT DETECTION.

6. View the results as the model is executing by going to the $JGAME/<scenario_name>/ debug directory. The computation debug statements are held in the files labeled **SIMU53<TimeStamp>.debug**.

Note this file will be closed and a new file opened each time a checkpoint occurs.

3.7 JTLS-2026-17664 Air Debug Going To Screen Not To File

If the user turns on the Air Debug Flag, some debug information is supposed to be placed in the Air Debug file. The output is not going to the file, it is going to the screen.

The problem was fixed.

While reviewing the code, it was noted that the current air debug information is not very useful since it prints out debug information for all flying air missions. What really is needed is to print out the debug information for the one air mission that is of interest to the Technical Control team attempting to analyze what is happening with a particular mission. The code was changed to only print out the debug information for the one air mission assigned as part of the DEBUG.OBJECT order.

3.8 JTLS-2026-17665 Debugging Cruise Missiles Did Not Work

Cruise Missile movements was designed to be a part of the Air Debug output. An error in the order file did not allow the Controller to specify a Cruise Missile that should be followed.

The error in the order panel's Cruise Missile specification field no longer worked. It was corrected to use a new methodology that was developed several years ago.

While testing this correction, it was determined that the debug information was not useful for Hypersonic Weapons. The debug information was updated to be more useful.

3.9 JTLS-2026-17669 Controller Political Country Update

When the Controller updates a Unit's political country, the JODA update for political country was not being sent. Additionally, only the Unit's HRUs were being updated, instead of HRUs, Targets owned by those HRUs, and the Unit's own owned/associated Targets.

Additionally, the symbol name update to the JODA was incorrect: symbol name updates were not being sent for HRUs, and both HRUs and Targets used an update type that was never sent to the JODA due to a missing call.

An additional JODA update for a Unit's political country was added. Two new routines were created to handle necessary updates required for all other associated objects (HRUs and Targets).

The JODA updates for HRUs now pass symbol names, and the update type was changed to correct the missing call for both HRUs and Targets.

3.10 JTLS-2026-17671 Crash Splitting All Ships In Formation

A user ordered all ships in a Formation to split off into a new Formation, crashing the model.

The design and order help text indicated that splitting all ships in a Formation into a new Formation is not legal. The code did not properly check for this order error. The check was added. The Split Formation task is now rejected if the user has ordered all ships to split into a new Formation.

3.11 JTLS-2026-17675 Password Reset Prompt Window Too Small

The password reset prompt was reported to be too small, as the initialized height of the window cut off the password requirement text.

The problem could not be replicated. As a work around, the initial height of the window was slightly increased.

3.12 JTLS-2026-17676 ICP Whip Side Columns Difficult To See

The WHIP side columns in the ICP have the header backgrounds set to the Force Side colors. The column background however are set to a default white color. When there are many Sides in a scenario, this can make clicking on the checkboxes in the columns difficult to see.

The column background colors have been changed to match the Side colors, exactly like the column headers. This makes differentiating columns easier to see visually.

3.13 JTLS-2026-17678 Cannot See Foreign Ships In Formation

A Blue Formation included Green ships and some Neutral ships. When asking to see the ships in a formation, from the Blue Side WHIP IMT, the user could not see the Green or Neutral ships.

A new IMT panel capable of showing foreign ships was created.

3.14 JTLS-2026-17681 Creating Naval Unit Aircraft Check Messages Bad

When a user creates a new Unit during game execution, two aircraft checks are accomplished:

- If the user specifies a type of aircraft to assign to the new Unit and the Unit prototype does not include any aircraft Combat Systems, a message is generated and the order is rejected.
- If the assigned prototype includes an aircraft Combat System and the user did not indicate the type of aircraft that should be assigned, a message is generated and the order is rejected.

In both situations, the generated message assumed that the Unit being created used a Tactical Unit Prototype (TUP). This is no longer true, because even Naval Units can be assigned aircraft.

The code was changed not to output the index identifier for the prototype, but the name of the prototype. In turn the message format file was changed not to assume that the index provided

belonged to a TUP, but simply printed out the name of the prototype that was included in the message data.

3.15 JTLS-2026-17686 Individual Target OPM Pages Missing HTML Extension

When opening an Online Player Manual (OPM) page for an individual Target from the WHIP, the URL for the target was missing the ".html" extension, causing the load of the page to fail. Each Target has an OPM page.

The ".html" extension was added to the URL.

3.16 JTLS-2026-17688 JDSP String Duplicate Routine

The routine set_STRING() allows the copying of memory from the source to the destination when the source and destination are the same memory location. For Linux, this is undefined and may not copy properly, which resulted in garbage in the destination string.

The set_STRING() routine was modified to first check if the source and destination were the same. If so, the copy is unnecessary.

3.17 JTLS-2026-17693 Allow Wide-Column IMT Searching

The current IMT searching capability only applies to a specific column. Users have to right-click the column header they want to search.

A search bar was added at the bottom of the IMT frame that allows users to enter a search query. Searching is applied to every visible column. The IMT table will update with all rows where there is a match.

3.18 JTLS-2026-17695 Prevent Orbit Paths With Identical Termination Points

Orbiting air missions with very close orbit path points (including start and end points) cause the model to perform air mission movement at a very high frequency. This causes a strain on the model.

If given orbit paths are too close to each other, either convert the order to Orbit Point or reject the order. The criteria for determining a close orbit path is all distances between each points within a given orbit path (including start and end points) being less than the defined threshold. The order is converted to Orbit Path if no given Orbit Path exists, and only start and end points are given.

The order is rejected if the Orbit Path is given, as the user's intention is more ambiguous in this case.

3.19 JTLS-2026-17696 Whip Notification Tool Updates

The WHIP Notification Tool (WNT) incorrectly displays Super WHIPs. The WNT displays all WHIP logins in a tree like structure where each WHIP is listed under its corresponding side. Super WHIPs have multiple Sides associated with them, and are displayed in the WNT only under one Side. This makes finding Super WHIPs difficult, as you need to search under each possible side the Super WHIP is assigned to in order to find it.

Additionally, the WNT does not have search capabilities for searching for WHIP logins.

The WNT now displays duplicate Super WHIP entries under each corresponding Side. When a user updates the state of the Super WHIP entry, any duplicate logins will also update accordingly.

A search bar was added to allow users to dynamically search for WHIP login names. The tree will update per keystroke.

3.20 JTLS-2026-17698 Red Background in Error Fields Are Too Harsh

The red background box color for error text fields make the foreground text difficult to see, especially for black text color.

The background color for those boxes was changed to light red for easier visibility of the text. The following components have been changed:

- WHIP: error text field search for IMT screens, Command Hierarchy, ORBAT search
- DDS: error text field search for ORBAT search, Command Hierarchy, Logistics Hierarchy, DJINN database search
- ICP: cell editor error message that pops up in the lower screen

3.21 JTLS-2026-17700 DDSC Header Titles For ICAO Fields

The header title for the ICAO fields in the Squadron and the FARP unit DDSC tables should be capitalized to "ICAO" to be consistent with the other table header titles.

The header title for the ICAO fields in the Squadron and FARP unit DDSC tables was capitalized.

3.22 JTLS-2026-17702 Leftover Line Grids Logic Error

A "Left Over Line Grids" logic error was continuously generated by the CEP.

When a Naval unit attempts to enter a sealane network, the logic finds the closest network node as an entry location. During this process, the logic calls a routine that determines whether a path to a node is feasible. However, that routine was not emptying the data used to determine the

path to a node that is being tested for feasibility, but the routine was expecting an empty data set. To stop the logic error, code was added to empty the data set after each call.

Note this correction also applied to Air Missions attempting to enter a flight path network.

3.23 JTLS-2026-17707 ICP Whip Panel Has No Searching Capability

Searching for a WHIP name in the ICP WHIP panel can be difficult when there are many WHIPs in the scroll panel.

A search bar was added to the lower panel to allow dynamic searching that updates the WHIP's table view per user keystroke.

3.24 JTLS-2026-17710 JODA Not Updated Removing Naval Support Unit

The JODA is not being updated if a WHIP operator changes the routine support unit for a Naval Unit to None.

The code that updates this JODA attribute was not being skipped in the new Support Unit was None. This error was corrected.

3.25 JTLS-2026-17712 Crash Magic Moving HRU On Grid Corner

The model crashed when a Controller submitted a Magic Move order for an HRU that was placed exactly on a grid corner. The crash occurred because the model was attempting to remove the HRU from the wrong grid.

This problem has happened before and was corrected, but the Magic Move routine was missed and did not have the correction that was previously implemented in other places when an HRU is being removed from a grid set.

The routine that caused the crash was missed because the code to remove an HRU from a grid was not centralized. Instead of correcting this one routine, the decision was to centralize the code and always use one routine to correctly remove an HRU from the grid to which it belongs.

3.26 JTLS-2026-17714 Changing SLP Convoy Limits Changes

A user wanted to change the attribute for a Barge Convoy, indicating that it should have a minimum distance of 1000 KM, rather than the current database value of 20 KM. When changing that parameter, the end result was that the SLP Barge parameter indicating that the Minimum Unit Size was blank. The model did not crash, but OPMs could not be generated.

The problem was that the code to interpret the Convoy limits did not check if the user had in fact changed the Minimum Unit Size parameter. This caused the Minimum Unit Size attribute to be set to "zero" because the order field was not filled.

Several issues were found as a result of fixing this issue:

- The Utility that passed the requested Supply Limit changes was changed to an "initialize" Utility. This results in the utility order being filled with negative 1, when the user does not fill in an optional field.
- The code to interpret the utility records was changed to use the coding standard of .ORDER.INITAIIZATION.VALUE indicating to everyone accessing the code to expect -1s in unfilled fields.
- The Show SLP parameter did not properly identify which Convoy type was being changed if the SHOW ALL option was set to No.

All three issues were fixed.

3.27 JTLS-2026-17719 Conflicting Mission Canceled Reports

When an air mission with a resource task that was delayed over its maximum delay time is canceled, the user received two conflicting error messages:

"The mission was canceled by a Player" and "The Mission was Canceled. Its Maximum Launch Delay Time was Exceeded. It was Delayed Due to Aircraft Availability."

The first message was incorrect and should not have been generated. This has been corrected.

3.28 JTLS-2026-17721 DDSC DRM Help Link Not Functioning

The DDSC includes a function that is intended to display the associated PDF file of the Data Requirements Manual. However, this help link to the DRM is currently not functioning.

An automated process that was used to split the Data Requirements Manual PDF file into separate files inadvertently deleted the "dds" documents subdirectory. This occurred due to a change in the name of the data_requirements_manual.pdf file. As a result, the JTLS-6.4.4.0.documents.tar.bz2 and JTLS-6.4.4.0.documents.zip files did not contain the "dds" documents subdirectory. The related cron job has been corrected, and the "dds" documents directory has been restored from the previous JTLS-GO 6.4.3.0 release.

3.29 JTLS-2026-17725 Login Build Default Context-Sensitive Menu Option

The Login Build order should be one of the default options in the Context Sensitive Menu on the Controller WHIP. Set Unit Disease Failures should be replaced, as it is rarely used.

The Set Unit Disease Failures order was replaced with Login Build in the Unit Context Sensitive Menu for Controller WHIPs.

3.30 JTLS-2026-17726 No Weapons Listed In Fire Artillery Order

A Fire Artillery order was selected for a Unit that had indirect fire combat systems. The order had no weapons in the Projectile Type drop-down list, even though the Unit had the correct ammunition supply categories, with non-zero amounts on hand.

This behavior was reproduced by creating a new artillery Unit while the game was executing. After the new Unit was TPFDDed into the game, the Fire Artillery drop-down list did not have any weapons. The model failed to create the list of usable weapons for the newly-arrived Unit.

The existing Unit arrival process built weapon lists for database Units arriving at game start but did not build lists for new Units created during execution. Because both situations execute the same Finish Unit Arrive routine, the routine call that builds the usable weapons list was moved from the Complete Unit Initialization routine (used only at start) to the Finish Unit Arrive routine.

3.31 JTLS-2026-17728 Incorrect Satellite Type Printed

A Controller created several satellites using the JTLS-GO Satellite Service (JSAT). All of the satellites were created with a particular Aircraft Target Class (ATC).

The satellites were then engaged using an ADA site. One of the satellites was destroyed and the Air Defense Report was generated. Part of this message describes the ATC for the satellite. In this case, the message reported it having the wrong ATC.

The Air Defense Report referenced the wrong data item. The data item was changed to match the output variable location in the write statement in the underlying routine.

3.32 JTLS-2026-17730 Change Fuel Type Crash

A Player sent a Change Mission Parameter order to change the fuel type of an air mission that was in Fuel Delay status, which caused a model crash.

A mission in Fuel Delay posture does not have an assigned fuel type. A request to change fuel type from the Change Mission Parameter order cannot be executed before the mission's initial fuel type has been assigned. The logic was changed to reject a change fuel type request when the mission is in Fuel Delay posture with an appropriate player message. This change is consistent with existing logic which also rejects a change fuel type request if the mission has already started (i.e. in the air).

3.33 JTLS-2026-17734 Create Unit Rejection Message Names

A unit was created with a long name and a short name. Another user then tried to create the same unit using the same long name and a different short name. A rejection message was received, saying the short name and the long name were already in use by another unit.

Creating a unit A, then creating a unit A[1] with the same long name but different short name, did not reproduce the message that both short and long name are already in use. The message correctly highlighted just the long name.

A further investigation revealed that it is possible for the new unit A[1] to have shared the short name with a completely different unit B. The message does not specify whether the unit that overlaps with the short name is the same as the unit that overlaps with the long name.

A new line was added to specify how many units overlap with the short and long names of the new unit. This detail removes the confusion on how many names overlap.

3.34 JTLS-2026-17737 Combined Aircraft Load Not Destroyed

In the routine Check Load Validity, a combine aircraft load is created which combines a user-specified load with the aircraft default load. This combined load is checked for validity that it will fulfill the requirements of the mission type. Once the check is done, the combined load is no longer needed. It was only destroyed and its memory released if the decision was that the load was valid. If the load was invalid, the load was not released.

The code was changed to always destroy the combined load, whether it was or was not valid.

3.35 JTLS-2026-17739 OPM Unit Page Link 16 Mission Blocks Link

A Unit's mission block data was missing from the OPM Unit page, as well as the corresponding list of Link 16 Mission Blocks page.

An OPM page for Unit mission blocks was added. A row was also added to the Unit page that links to the Unit's mission block.

3.36 JTLS-2026-17742 Contamination Status Incorrect

The model establishes the status of Airbases, FARPs, and Naval Units to report to C4I systems. One of the checks is whether the unit is within a contamination area. The distance between the unit and the weapon explosion was not properly checked.

The distance check was corrected.

3.37 JTLS-2026-17744 Inefficient End Contamination Logic

When nuclear or chemical contamination ends, the model does one last contamination assessment for the unit. After the assessment is complete, the model need to determine if the unit is covered by other contaminations of the same type. This determination is inefficient and ends up cycling through each end contamination event in the game. It only needs to check those end contamination events that are covering the unit.

The logic was changed to check only contamination events that are currently covering units that were covered by the one contamination event that just finished.

3.38 JTLS-2026-17748 End Contamination Does Not Save Radius

Event attributes are saved with values passed in during event scheduling. The radius of a contamination area is passed in on every scheduling instance, but the event routine END.CONTAMINATION itself did not have it defined as its parameter.

The event routine was using the Targetable Weapon's radius instead. The Manage Contamination Area order allows a user-defined radius of a contamination area, making the weapon radius incorrect. The parameter for event attribute radius was added to the END.CONTAMINATION routine. Redundant lines for manually setting event attributes were deleted. In JDS routine, the line for accessing weapon radius was replaced with EC.Radius.

Also, there was an error in which a checkpoint restart would not initialize more than 1 dispersal object tied to END.CONTAMINATION event. This was due to its reference number being passed in as a pointer, instead of a value. This was fixed.

3.39 JTLS-2026-17750 JDS Update Object Loses Weapon Count

If the user performs a JDS Update on a Unit, the Unit loses all of its Weapon Count data.

The routine JDS.UNIT.REINITIALIZATION is responsible for deleting any sub-entities of a Unit and the Unit itself before initializing everything in JODA. It did not delete Weapon Count objects, so they existed in the CEP and JODA. The JXSR does not link the existing Weapon Counts when a Unit is created after them.

New Weapon Counts associated to the Unit are now created after the Unit is created. The reinitialization routine was also updated. No other changes were required as the routine already takes care of the weapon initialization.

3.40 JTLS-2026-17751 Add Orbit Task To Air Mission Rejected

A Patrol air mission was guiding off a Formation. The Player submitted a Manage Air Mission Tasks order to add a new orbit task at a different bearing and range. The order was rejected with a peculiar message:

MANAGE.AIR.MISSION.TASKS order named ORBIT cannot be executed:

The specified HRU MH60R-060003 does not exist.

The rejection error was caused by an overloaded variable in the Manage Air Mission Tasks order. The Formation name and a Unit name used the same variable, which was then accessed by the underlying routine. The logic assumed that the variable was a Unit name, when in fact it was a Formation name. The logic attempted to accept the order as a Unit and when it determined no

such Unit existed, it assumed the Unit was actually an HRU. The logic then determined that the HRU did not exist and rejected the order.

The problem was corrected by assigning the Formation name to its own variable in the Manage Air Mission Tasks order. Also, the routine was changed to look for the Formation name in the correct order field when processing a Manage Air Mission Tasks order.

3.41 JTLS-2026-17759 Satellite Fire Weapon Multiple Received Orders

A Satellite Fire Weapon order was submitted to an orbiting satellite that was at the end of its programmed route and no longer moving. The CEP window filled with repeated Received Order events.

When the satellite first received the Satellite Fire Weapon order, it was not within range of a satellite communications facility and the owning force side did not have a global satellite communications capability. By design, the Satellite Fire Weapon order was rescheduled to be received after the satellite's next move in case it acquired communications. Hours later, the satellite reached the end of its route and stopped moving without ever acquiring communications. The Satellite Fire Weapon order event was then rescheduled every few seconds because the satellite's next move time was forever froze,n and therefore earlier than the current advancing game time.

New logic was added to cancel the Satellite Fire Weapon order when a satellite stops moving and communications are not transmittable to the satellite. A new message was added to inform the Player of the situation.

Another message was added to inform the Player the first time (only) a Satellite Fire Weapon order is submitted and there are no available communications to the satellite. In this situation, the Player is informed that the model will keep rescheduling the order receipt after each move until the satellite has communications and can fire.

4.0 REMAINING ERRORS

Every effort has been made to correct known model errors. All reproducible errors that resulted in CEP catastrophic software failures (crashes) have been corrected. Other corrections were prioritized and completed according to their resource cost-to-benefit relationship.

The following list of issues is known and have not been fixed in time to make it into this release of JTLS-GO 6.3.10.0.

4.1 DDSC/WHIP/JOBE - CADRG Map Zoom

When using the CADRG map projection, if the width of the map is less than the height, the zoom tool does not work correctly.

4.2 MHE Targets Loading Air Mission Can Cause a Crash

MHE targets should be avoided for loading and unloading air missions. It is suggested that the database be set to “Do Not Use” for Air Missions.

4.3 JTLS-GO Strategic Lift Missions Are Not Working Properly

Strategic Lift Missions, used to move TPFDD assets into the Theater and report the results to a real-world TPFDD processing system, has not been updated to work within JTLS-GO 6.3.

4.4 Tactical Ground Formation Attacks Do Not Work

The ability to send a Tactical Ground Formation on an Attack mission has been temporarily disabled due to reliability issues.

4.5 ATOT Spreadsheet Lacks Detailed Field Checking

The ATOT Spreadsheet Parser has been found to have numerous issues within the Spreadsheet format that are not caught and cause the spreadsheet parser to crash. Fixing the uncovered issues are being worked and should be fixed prior to the next maintenance release of the JTLS-GO 6.3 series.

4.6 Moving Combat System Supplies Can Reduce Unit Strength To Zero

If a user does a mandatory transfer of Combat System supplies from one unit to another, the providing unit can be emptied out and exists without any Combat Systems or personnel. This situation needs to be thoroughly and properly handled.

4.7 Upgrade Procedures For Maintenance Release of PostgreSQL

The Development Team continues to look for viable options to upgrade from one maintenance release of PostgreSQL to a newer maintenance release.

APPENDIX A. ABBREVIATIONS AND ACRONYMS

Terms are included in this Appendix to define their usage in JTLS-GO design, functionality, and documentation.

AAA	Anti-Aircraft Artillery
AADC	Area Air Defense Commander
AAL	Air-to-Air Lethality
A/C	Aircraft
ACP	Air Control Prototype
ADA	Air Defense Artillery
AEW	Airborne Early Warning
AFB	Air Force Base
AG	Air-Ground (Air-to-Ground)
AI	Air Interdiction
AIM	Air Intercept Missile
AIREF	Air Refueling
AKL	Area Kill Lethality
AMMO	Ammunition
AO	Area of Operations
AOC	Air Operations Center
APC	Armored Personnel Carrier
ARECCE	Armed Reconnaissance
ARTE	Air Route
ARTY	Artillery
ASC	Automatic Supply Calculation
ASCII	American Standard Code for Information Interchange
ASW	Anti-Submarine Warfare
ATC	Aircraft Target Category
ATGM	Anti-Tank Guided Missile
ATK	Attack
ATO	Air Tasking Order
ATORET	Air Tasking Order Retrieve Program
ATOT	Air Tasking Order Translator
AWACS	Airborne Warning And Control System
AZ	Altitude Zone

BADGE	Bilateral Air Defense Ground Environment (used by Japan Defense Agency)
BAI	Battlefield Air Interdiction
BDA	Battle Damage Assessment
BDE	Brigade
BN	Battalion
C3	Command, Control, and Communications
C3I	Command, Control, Communications, and Intelligence
C4I	Command, Control, Communications, Computers, and Intelligence
CA	Civil Affairs
CADRG	Compressed ARC Digitized Raster Graphics
CAP	Combat Air Patrol
CAS	Close Air Support
CAT	Category
CCF	Central Control Facility
CCP	Command Control Prototype
CEP	Combat Events Program
CMDR	Commander
COP	Common Operational Picture
CP	Combat Power
CS	Combat System
CSP	Combat System Prototype
CTAPS	Contingency Tactical Air Planning System
CTG	Commander Task Group
CTRL	Control keyboard command
DCA	Defense Counter Air
DCL	Digital Command Language
DDS	Database Development System
DEMSDB	Demonstration Standard Database
DISA	Defense Information Systems Agency
DIV	Division
DMA	Defense Mapping Agency
DoD	Department of Defense
DOS	Days of Supply
DPICM	Dual Purpose Improved Conventional Munitions

DS	Direct Support
DSA	Directed Search Area
DTG	Date Time Group
EC	Electronic Combat
ECM	Electronic Counter Measure
ECP	Engineering Change Proposal
EEI	Essential Elements of Information
ELINT	Electronic Intelligence
ELS	Entity Level Server
EODA	Entity Level JTLS Object Data Authority
ETA	Estimated Time of Arrival
FARP	Forward Arming and Refueling Point
FLP	Fire Lethality Prototype
FLOT	Forward Location of Troops
FOL	Forward Operating Location
FWL	Frederick W. Lanchester (originated a differential equation model of attrition)
GAL	Gallon
GCCS	Global Command and Control System
GRTE	Ground Route
GS	General Support
GSR	General Support Reinforcing
GUI	Graphical User Interface
HARM	High-speed Anti-radiation Missile
HE	High Explosive
HELO	Helicopter
HMMWV	High Mobility Multipurpose Wheeled Vehicle
HQ	Headquarters
HRU	High Resolution Unit
HTML	Hypertext Markup Language
HTT	High resolution unit Target Type
HUP	High resolution Unit Prototype
ICM	Improved Conventional Munitions
ICP	Interface Configuration Program
ICPLogin	Interface Login Program

ID	Identifier
IFF	Identification Friend or Foe
IIP	Intelligence Information Prototype
IMT	Information Management Tool
INFO	Information
INTEL	Intelligence
JCATS	Joint Conflict And Tactical Simulation
JDA	Japan Defense Agency
JDPI	Joint Desired Point of Impact (formerly DMPI: Desired Mean Point of Impact)
JDS	JTLS Data System
JDSP	JTLS Data System Protocol
JEDI	JODA Entity Data Identifier
JMCIS	Joint Maritime Combat Information System
JMEM	Joint Munitions Effectiveness Manuals
JODA	JTLS Object Distribution Authority
JOI	JTLS Operational Interface
JPL	Jet Propulsion Laboratory
JRSG	Joint Rapid Scenario Generation (formerly JIDPS: Joint Integrated Database Preparation System)
JSDF	Japanese Self-Defense Force
JTLS	Joint Theater Level Simulation
JTLS-GO	Joint Theater Level Simulation - Global Operations
JTOI	JTLS Transaction Operational Interface
JXSR	JTLS XML Serial Repository
KIA	Killed In Action
KM	Kilometer
KNOTS	Nautical miles per hour
LA	Lethal Area
LAN	Local Area Network
LAT	Latitude
LB	Login Build (JTLS order type)
LDAP	Lightweight Directory Access Protocol
LDT	Lanchester coefficient Development Tool
LOG	Logistics
LOGIN	Logistics Input

LOGREP	Logistics Report
LONG	Longitude
LOTS	Logistics Over The Shore
LR	Long Range
M&S	Modeling and Simulation
MAPP	Modern Aids to Planning Program
MB	Megabyte
MCP	Mobility Counter-mobility Prototype
MCR	Model Change Request
MG	Machine Gun
MHE	Material Handling Equipment
MIP	Model Interface Program
MOGAS	Motor Gasoline
MOPP	Mission-Oriented Protective Posture
MOSAIC	NCSA user interface software
MOTIF	X Window System graphical interface
MP	Maneuver Prototype
MPP	Message Processor Program
MSC	Major Subordinate Command
MSG	Message
MTF	Message Text Formats
MUREP	Munitions Report
MUSE	Multiple Unified Simulation Environment
NCSA	National Center for Supercomputing Applications (University of Illinois)
NEO	Noncombatant Evacuation Operations
NFS	Network File Server
NGO	Non-Governmental Organization
NIS	Network Information Service or Network Information System
NM	Nautical Mile
NTSC	Naval Telecommunications System Center
OAS	Offensive Air Support
OBS	Order of Battle Service (formerly UGU: Unit Generation Utility)
OCA	Offensive Counter-Air
OJCS	Organization of the Joint Chiefs of Staff

OMA	Order Management Authority
ONC	Operational Navigation Chart
OPM	Online Player Manual
OPP	Order Preprocessing Program
OTH	Over The Horizon
OTH Gold	Over The Horizon message specification
OTH-T	Over The Horizon-Targeting
pD	Probability of Detection
pE	Probability of Engage
pH	Probability of Hit
pK	Probability of Kill
PKL	Point Kill Lethality
POL	Petroleum, Oil, and Lubricants
POSIX	International operating system standard based on System V and BSD
PPS	Postprocessor System
PSYOPS	Psychological Operations
RAM	Random Access Memory
RDMS	Relational Database Management System
RECCE	Reconnaissance (air missions)
RECON	Reconnaissance (ground missions)
REGT	Regiment
RNS	Random Number Seed
ROE	Rules Of Engagement
RPT	Report
RSP	Reformat Spreadsheet Program
SAL	Surface-to-Air Lethality
SAM	Surface-to-Air Missile
SAM/AAA	Surface-to-Air Missile/Anti-Aircraft Artillery
SC	Supply Category
SCP	Simulation Control Plan
SDB	Standard Database
SDR	Scenario Data Repository
SEAD	Suppression of Enemy Air Defense
SIMSCRIPT	Simulation programming language (product of CACI, Inc.)

SIP	Scenario Initialization Program
SITREP	Situation Report
SLP	Sustainment Log Prototype
SOF	Special Operations Forces
SP	Survivability Prototype
SQL	Structured Query Language
SR	Short Range
SRP	Start/Restart Program (a JTLS component)
SRTE	Sea Route
SSM	Surface-to-Surface Missile
STR	Software Trouble Report
SUP	Ship Unit Prototype
SVP	Scenario Verification Program
SYNAPSE	Synchronized Authentication and Preferences Service
TADIL	Tactical Digital Interface Link
TCP/IP	Transmission Control Protocol/Internet Protocol
TEL	Transporter Erector Launcher
TG	Target entity attribute prefix
TGS	Terrain Generation Service (formerly TPS:Terrain Preparation System)
TGT	Target
TMU	Terrain Modification Utility
TOE	Table of Organization and Equipment
TOT	Time Over Target
TOW	Tube-launched Optically-tracked Wire-guided missile
TPFDD	Time-Phased Force Deployment Data
TTG	Target Type Group
TTL	Target Types List
TUP	Tactical Unit Prototype
TW	Targetable Weapon
UBL	Unit Basic Load
UIM/X	GUI builder tool
UNIX	POSIX-compliant operating system
UNK	Unknown
UOM	Unit Of Measure

USA	United States Army (U.S. and U.S.A. refer to United States and United States of America)
USAF	United States Air Force
USCG	United States Coast Guard
USMC	United States Marine Corps
USMTF	United States Message Text Format
USN	United States Navy
UT	Unit entity attribute prefix
UTM	Universal Transverse Mercator
VIFRED	Visual Forms Editor
VMS	Virtual Memory System
VTOL	Vertical Take-Off and Landing aircraft
WAN	Wide Area Network
WDRAW	Withdraw
WEJ	Web Enabled JTLS
WHIP	Web Hosted Interface Program
WIA	Wounded In Action
WPC	Warrior Preparation Center
WPN	Weapon
WT	Weight
WW	Wild Weasel
XMS	XML Message Service

APPENDIX B. Version 6.3.10.0 DATABASE CHANGES

Due to changes made in support of STR JTLS-2025-17024, the JTLS-GO 6.3.4.0 database format was changed to add a new column to the text_symbol_lu table, related to unit symbols.

This change requires that users **unload** their scenarios prior to installation of JTLS-GO 6.3.4.0, and then **load** them following installation, if users are upgrading to JTLS-GO 6.3.10.0 from JTLS-GO 6.3.3.0 or earlier.

APPENDIX C. VERSION Version 6.3.10.0 REPOSITORY CHANGES

No significant changes have been made to the structure of the JTLS-GO 6.3.10.0 repository.