JTLS-GO

Version Description Document

June 2025

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

JOINT THEATER LEVEL SIMULATION - GLOBAL OPERATIONS
(JTLS-GO 6.2.10.0)

[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, and Non-Governmental Organization (NGO) functions within a combined joint and coalition environment.

This *JTLS-GO Version Description Document (VDD)* describes the new features of the Version 6.2.10.0 delivery of the configuration-managed JTLS-GO software suite.

JTLS-GO 6.2.10.0 is the final Maintenance release of the JTLS-GO 6.2 series that includes a repository of standard data, a demonstration scenario based in the western Pacific. No further updates to the JTLS-GO 6.2 series will be released. No new system enhancements have been added as a result of this release. 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 Avenue, Suite A
Pacific Grove, California 93950
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 6.2.10.0 of the configuration managed Joint Theater Level Simulation - Global Operations (JTLS-GO®) software suite. JTLS-GO 6.2.10.0 is a Maintenance delivery for the JTLS-GO 6.2 series of releases.

JTLS-GO 6.2.10.0 includes the entire JTLS-GO suite of software, a repository of engineering level data, and a realistic demonstration scenario based on the Western Pacific theater of operations called “wespac62”. Chapter 3.0 describes the software errors that have been fixed since the last release of the JTLS-GO 6.2 series.

JTLS-GO 6.2.10.0 executes on the Red Hat Enterprise Linux Server Version 8.7 64-bit operating systems. The Web-Hosted Interface Program (WHIP®) user workstation interface can be executed on any 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 6.2.8.0)
- *JTLS-GO Executive Overview* (JTLS-GO Document 02, Version 6.2.0.0)
- *JTLS-GO Configuration Management Plan* (JTLS-GO Document 03, Version 6.2.0.0)
- *JTLS-GO Controller Guide* (JTLS-GO Document 04, Version 6.2.7.0)
- *JTLS-GO Director Guide* (JTLS-GO Document 07, Version 6.2.0.0)
- *JTLS-GO Installation Manual* (JTLS-GO Document 09, Version 6.2.5.0)
- *JTLS-GO WHIP Training Manual* (JTLS-GO Document 10, Version 6.2.2.0)
- *JTLS-GO JOBE Quick Start Guide* (JTLS-GO Document 11, Version 6.2.2.0)
- *JTLS-GO Player Guide* (JTLS-GO Document 12, Version 6.2.3.0)

- *JTLS-GO Repository Description* (JTLS-GO Document 14, Version 6.2.0.0)
- *JTLS-GO Software Maintenance Manual* (JTLS-GO Document 15, Version 6.2.3.0)
- *JTLS-GO Technical Coordinator Guide* (JTLS-GO Document 16, Version 6.2.5.0)
- *JTLS-GO Entity Level Server User Guide* (JTLS-GO Document 19, Version 6.2.0.0)
- *JTLS-GO Federation User Guide* (JTLS-GO Document 20, Version 6.2.0.0)
- *JTLS-GO C4I Interface Manual* (JTLS-GO Document 21, Version 6.2.5.0)
- *JTLS-GO DoD Architecture Framework* (JTLS-GO Document 22, Version 6.2.0.0)
- *JTLS-GO DDS Training Manual* (JTLS-GO Document 23, Version 6.2.3.0)
- *JTLS-GO Air Services User Guide (*JTLS-GO Document 24, Version 6.2.0.0)

1.2.3 Updated Documents

- *JTLS-GO Data Requirements Manual* (JTLS-GO Document 05, Version 6.2.10.0)
- *JTLS-GO DDS User Guide* (JTLS-GO Document 06, Version 6.2.10.0)
- *JTLS-GO Version Description Document* (JTLS-GO Document 17, Version 6.2.10.0)

1.2.4 New Documents

No new documents are delivered with this version of the software.

1.2.5 Delivered Software Components

JTLS-GO 6.2.10.0 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)
- 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 188 and has not been changed as a result of this release.

- Web Services Manager (WSM)
- Web-Hosted Interface Program (WHIP) and its component programs:

  Apache Server (APACHE)

  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)
- 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.

Instructions for installing JTLS-GO 6.2.10.0 are provided in the *JTLS-GO Installation Manual*. Installing a previous version of JTLS-GO prior to installing JTLS-GO 6.2.10.0 is not necessary. The software provided with this delivery is a complete release that includes all files and code required to execute JTLS-GO.

The basics of installation have not changed significantly, but due to many Cyber-Security improvements, a new Linux RPM package named “xerces-c”, is required to run JTLS-GO 6.2.2.0 and all later versions. The Synapse will not function without this package. Prior to installing JTLS-GO 6.2.10.0, please run the RPM checking script delivered with JTLS-GO to ensure that this package and all other packages are installed as part of your Linux operating system.

Due to repeated problems at exercises, JTLS-GO Version 6.2.7.0 changed the method the ICP uses to save WHIP passwords. This improvement resulted in a version change to a scenario’s ICP database files. If you did not upgrade to JTLS-GO Version 6.2.7.0, please refer to the complete explanation and instructions on how to alter existing scenarios to use the new ICP database structure and format in Chapter 2 of the *JTLS-GO Version Description Document* for Version 6.2.7.0. This document is delivered with JTLS-GO 6.2.10.0 in the $JTLSHOME/documents directory.

During installation of JTLS-GO 6.2.10.0, users must perform the following steps:

1. **Unload** their scenarios in their current version of JTLS-GO 6.2 (prior to installing JTLS-GO 6.2.10.0).
2. Install JTLS-GO 6.2.10.0, following the instructions in the *JTLS-GO Installation Manual*.
3. Run the **mend-db-rns** script, located in the $JTLSHOME/script directory, for each of their scenarios.
4. **Load** their scenarios, and then immediately **unload** them.

These instructions are due to changes made in JTLS-GO 6.2.5.0, related to random number seeds, as documented in STRs JTLS-2024-16756, JTLS-2024-16758, and JTLS-2025-17069.

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 “repository62”. 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 “wespac62”, which is suitable for training and demonstrations.

1.3 INTERFACE COMPATIBILITY

1.3.1 Support Software

JTLS-GO 6.2.10.0 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 8.7, 64-bit architecture.

  JTLS-GO 6.2 has been tested with the following versions of Linux 8:

  RedHat Linux 8.7 - this operating system license must be purchased.

  Oracle Linux 8.7 - 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 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 have a Java-enabled web browser. JTLS-GO 6.2.10.0 has been tested on the following operating systems:

  Red Hat Linux Enterprise Server Edition Version 7.9, 8.4, and 8.7

  Oracle Linux 8.4 and 8.7

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

- JTLS-GO 6.2.10.0 is delivered with Apache 2.4.62, the latest available Apache security patch release.

- JTLS-GO 6.2.10.0 is delivered with the Adoptium project Temurin Java Development Kit (JDK) 1.8 Update 452 package, which is equivalent to the current version of OpenJDK.

- JTLS-GO uses IcedTea to provide the Java Web Start capability that implements the web-enabled JTLS-GO functionality. JTLS-GO supports IcedTea version 1.8.8.

- JTLS-GO database tools require a certified PostgreSQL 11.19 database server and the full PostgreSQL installation. A containerized solution, that fulfills this specification, is provided as part of the JTLS-GO download. It is not necessary to use the delivered containerized solution, but it is the easiest method to meet the requirements of JTLS-GO 6.2.10.0. There are several alternative methods available for obtaining the PostgreSQL 11.19 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.

- TCP/IP is required for inter-process communication between the JODA data server and all user interface programs. The version of TCP/IP included with the supported versions of Red Hat Linux ES is sufficient.

- 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.

- 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.

- 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.

- JTLS-GO 6.2.10.0, 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 8.7 using gcc (GCC) 8.5.0 20210514 (Red Hat 8.5.0-15.0.2).

  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 Apache, PostgreSQL IceTea, and Java.

JS/J7 Cybersecurity branch has approved the following update procedure. On a quarterly basis, a maintenance release of JTLS-GO is produced which includes the latest security patches for Java 1.8, PostgreSQL 11, IceTea 1.8, and Apache 2.4. To remain compliant with these approved procedures, users should upgrade and use the latest maintenance release of JTLS-GO.

Please contact the U.S. Government Program Manager, Mr. Douglas Failor, at douglas.l.failor.civ@mail.mil to obtain the completed Cybersecurity paperwork and a current Gate completion certificate.

Note: PostgreSQL 11 is no longer supported by the PostgreSQL organization. JTLS-GO 6.2.10.0 is being delivered with the last PostgreSQL 11 security release from November 2023. The JTLS-GO 6.3 series has moved to the newest supported version of PostgreSQL, but due to some major changes in PostgreSQL it is impossible to use this newer version of PostgreSQL with the JTLS-GO 6.2 series. If this situation causes issues gaining local cyber approval to run JTLS-GO 6.2.10.0, an organization has no choice but to move to the JTLS-GO 6.3 series.

As a result of new security requirements built into JTLS-GO 6.2.0.0 as part of ECP JTLS-2022-15976 “Encrypt Passwords To Start WHIP/DDSC”, users must delete **all** pre-JTLS-GO 6.2 scenarios from their $JGAME directory. This will require you to set up your scenarios from a fresh state:

1. Convert your scenarios to JTLS-GO 6.2.0.0 using the Version Conversion Program (see Chapter 13 of the *JTLS-GO DDS User Guide* for instructions).
2. Perform Option 3, “Setup System For A Specific Scenario”, for each scenario.
3. Perform Option 5, “Run Interface Configuration Program”, for each scenario.

This is **not** a requirement if you have already installed JTLS-GO 6.2.0.0 and are upgrading to a JTLS-GO 6.2.n.0 maintenance release.

1.3.3 JTLS-GO High Level Architecture Compliance

The JTLS-GO 6.2.10.0 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, Federation testing of JTLS-GO 6.2.0.0 with CAE's GESI model has not been accomplished. CAE should be contacted concerning the continued support of GlobalSim.

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 6.2.10.0 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

Several database structure differences exist between JTLS-GO 6.2 series and the previous JTLS-GO 6.1 series database structure.

To upgrade your JTLS 6.1 scenario to JTLS-GO 6.2 compatibility, see instructions listed in the *JTLS-GO DDS User Guide*, Chapter 3.1, followed by the instructions in Chapter 13 of the *JTLS-GO DDS User Guide.*

> Users should download and re-load their scenarios into PostgreSQL, due to changes made in STR JTLS-2023-16290 PSQL Statement Fails For SVP Warning Correction, included in JTLS-GO 6.2.2.0, which was released in July 2023. This will recreate the related check constraints in the database schema. This procedure only needs to be execute once for every scenario. If the procedure was followed after the release of JTLS-GO 6.2.2.0, it does not need to be followed again,

1.4.1 JTLS-GO Symbol Set

Over the past several years, the Database Team has added and updated the Default Symbol Set used for the delivered scenarios. An organization is not required to use the JTLS-GO Default

Symbol Set, but If there is a desire to do so, the following information is provided to help determine how to use the JTLS-GO Default Symbol Set improvements and changes that have been made.

With the release of JTLS-GO 6.2.5.0, STR JTLS-2024-16648 Bad Symbol ID Code From JSYMS resulted in several changes to the Default Symbol Set. There are two possible situations in which a user organization can currently stand with respect to the use of the JTLS-GO Default Symbol Set. The method that should be used to update a scenario's Symbol Set to the latest Default Symbol Set is included for each situation. Please see the *JTLS-GO 6.2.5.0 Version Description Document* for further information.

1.4.1.1 User Organization Using Pre-JTLS-GO 5.0.0.0 Legacy Default Symbol Set

To change a scenario's Symbol Set under this situation, prior to unloading your JTLS-GO 6.2.0.0 formatted data from your PostgreSQL database server into the JTLS-GO 6.2.0.0 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.

1.4.1.2 User Organization Using Post-JTLS-GO 6.0 Symbol Set

The symbol 2525 ID Codes, which are used by C4I systems to identify the type of object, were expanded and refined as part of this JTLS-GO 6.2.5.0 release. If you have scenarios based on the JTLS-GO repository symbols, you are encouraged to update the symbol sets for your organization's scenarios. This can be accomplished by executing the following steps after the installation of JTLS-GO 6.2.5.0:

1. Conduct a download of the scenario.

2. Go to the $JDATA/scenario/<scenario_name>/symbols directory by entering the following command:

   ```
   cd $JDATA/scenario/<scenario_name>/symbols
   ```

3. Compare your symbol.scf file to the JTLS 6.2.5.0 version under the repository62 scenario by enter the following command:

   ```
   diff symbol.scf $JDATA/scenario/repository62/symbols/symbol.scf_jtls60
   ```

4. If the names of symbols in the scenario symbol file are identical to the repository symbol file or if your scenario symbols are a subset of the repository62 version, update your symbol file by entering the following command:

   ```
   cp $JDATA/scenario/repository62/symbols/symbol.scf ./symbol.scf
   ```

Your symbol set will now be updated. For safety and to ensure Step 4 is done correctly, do a load of the scenario. Check the log files to ensure there was a clean load

5. If your symbol file has additional symbols not found in the current repository62 version, you can still update your symbol file, but will need to perform a manual integration of the symbol.scf file.

   Bring up the jsyms program for your scenario, by selecting Option 1 “Prepare or Alter a Scenario Database”, and then Option 5 “Configure Symbols”, from the JTLS Menu.

6. Perform a save in the jsyms program.

- If the scenario is loaded on the DDS, select “Yes” to fully update the symbols in the scenario.

- If your scenario is not loaded into the DDS, select “No” to simply update the ASCII version of the scenario’s usable graphics symbol located in the <scenario_Name>.gs file. The next time you perform a load, the most current updated symbols will be loaded from the <scenario_name>.gs file.

7. Exit the jsyms program.

1.4.2 Standard Repository Changes

The JTLS-GO Database Team has continued to improve and expand the unclassified JTLS-GO data repository, named to “repository62”. 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.

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 11.19 required to operate JTLS-GO 6.2.10.0.

2.0 ENGINEERING CHANGE PROPOSALS

No new model capabilities were added to JTLS-GO 6.2.10.0 as a result of implementing authorized minor Engineering Change Proposals (ECPs).

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-2025-17072 SVPR Error 451 Correction Does Not Work

In the SVP, Error 451 has two corrections. The correction offering to "blank out the track blocks" for a squadron has no effect on the existing track blocks.

The error was checking data in two different tables to see if a squadron had JU numbers or source numbers and, if it did, whether it was assigned to a Link 16 network. If there was an error, the correction assumed the error was generated solely based on the data in the squadron table. In fact the error was also generated if the squadron had source JU numbers in the Squadron JU table and no Link 16 network was assigned.

Instead of confusing the user by combining corrections to two different tables into Error 451, the errors associated with the Squadron JU number table were moved to new Error 422.

3.2 JTLS-2025-17074 Leaflet OAS Mission Does Not Fly

An Orbiting Air Support (OAS) order was submitted to direct an air mission to drop leaflets (1000KEAFLET.BOX) on a ground Unit. The mission entered Launch Capable posture, but immediately completed with a message saying "the mission is not carrying the right weapon to use. None of the weapons are effective against the specified target."

To use leaflets, the supply category for the weapon must have a special capability of LEAFLET. In the database, the 1000LEAFLET.BOX targetable weapon uses the supply category LEAFLETS-1000LEAF.BOX which correctly has a special capability of LEAFLET. Also, the 1000LEAFLET.BOX weapon has a surface kill lethality of NON.LETHAL_SKL, with zero Probability of Hit (PH) and zero Probability of Kill (PK) tables.

The problem was in the logic that determines which weapons can be fired. Any weapons with a PH and PK of zero were deemed "ineffective". If no weapons were effective, the mission was immediately terminated before launch.

To fix this problem, the code was changed to allow weapons with a special supply capability of LEAFLET to be considered effective, regardless of PH and PK values.

3.3 JTLS-2025-17078 Total Target Track Numbers not counted correctly

The SVP is miscounting the available Target Track numbers. For example, the SVP says 06001 to 06005 has four available numbers, when the correct number is five.

The logic in the code failed to allow for inclusive numbers. 06001 to 06005 when subtracted is four, then one should be added to allow for inclusive numbers.

While correcting and testing, it was also found that a Link 16-capable Target, where the owning and associated Units are the same, was counted as two Targets, needing two separate Target Track Numbers. This was also corrected.

3.4 JTLS-2025-17080 Incorrect Use of Sensor Type

In the logic that updates grid effects for a moved sensor target, the sensor type was mapped to a local variable, THE.TYPE, as .ENTITY.ST, .ENTITY.JT, or .ENTITY.AD. THE.TYPE was then passed to routines that created or moved grid effects as appropriate.

Later, the logic incorrectly used THE.TYPE again as the sensor target's use: .PASSIVE.SONAR, .ACTIVE.SONAR, or .SURFACE.SEARCH. The sensor's use was needed to determine the appropriate routines to check detections. Using THE.TYPE in this manner was the wrong context and removed the opportunity to detect surface or subsurface vessels.

The local variable THE.TYPE was replaced in the second context with the actual use of the sensor class (the sensor subcategory) so that the appropriate detection routines are called. In addition, THE.TYPE was misapplied to determine a Jammer's use. THE.TYPE was replaced with the actual use of the jammer class (the jammer subcategory).

3.5 JTLS-2025-17083 Inconsistent Random Number Seeds In Scenario

The ELS utilized several random number seeds for building initial templates and during execution. The names of these seeds changed in JTLS-GO 6.3, but not in JTLS-GO 6.2. Due to inconsistencies in the names of the seeds, users were unable to load their JTLS-GO 6.2.9.0 scenarios into the database.

The DDS scripts were missing the names of the random number seeds used by the ELS. These seed names were added and initialized in those scripts.

3.6 JTLS-2025-17091 Patch For Complete Naval Task

There was a crash in Complete Naval Task. The ship that was supposed to be completing a Naval Mine Laying task had no tasks in its task list.

The exact cause of crash could not be determined. A patch was added to the code to check for the situation. It is possible that this was caused by the misuse of the Correct Object Task Order. STR JTLS-2025-17171 Make Correct Object Task Critical Order has altered the order to make it easier to determine if the misuse of the order is causing the problem.

3.7 JTLS-2025-17092 Mission Adopts Intercept Speed Before Committing

An air mission increased its speed to maximum before being committed to an intercept.

In the logic that determines whether the mission can legally fire air-to-air weapons, a code error permitted the mission to speed up in certain situations, even though it was not currently intercepting. The error was corrected to first make sure the mission was currently intercepting before adjusting the mission's speed.

3.8 JTLS-2025-17097 AARC Memory Leak While CEP Is Connected

The AAR Client (AARC) service consumes memory while processing data from the JODA, which flows to the AARC while the CEP is connected. It normally requests and then returns the memory as it processes the data. However, even when the CEP is idle, the AARC increases its requirement for system memory without reducing its consumed memory.

The AARC periodically gains and removes objects in local memory as part of its main loop. A routine used for removing objects will request a list to hold all the objects prior to checking each object. When the routine finished with the list it was not returning the memory used by the list. This has been corrected.

3.9 JTLS-2025-17101 Allow Comparison Of Aircraft Type To Value Zero

If the user wants to Attach two units that own aircraft, both the attaching and accepting unit must own aircraft of the same type (such as "F16" or "C130"). The Attach order XML file represents this as a field restriction, where either the attaching unit or accepting unit's aircraft type must be zero (meaning the unit does not have an aircraft type), or the attaching unit aircraft type is equal to the accepting unit aircraft type.

The Order Management Authority (OMA) rejected the order if one of the units had an aircraft type but the other did not. The OMA knew the aircraft type as a JODA Entity Data Identifier (JEDI), but the value of zero was an integer, so the OMA could not compare the two values because they are different types of objects.

Code was added to check if the first value of a binary comparison is a JEDI and the second is an integer, and if so, to treat the second value as a JEDI instead, which allows the OMA to compare the objects as like types.

A minor spelling error was corrected in the order file for the help text informing the user of the problem when the unit aircraft types did not match.

3.10 JTLS-2025-17103 Create Squadron Without Home Base Crash

If a user creates a squadron during the game, and does not assign it a home base or alter its location as part of the TPFDD order to bring the unit into the game, the model crashes.

The crash happens because the unit has a legal latitude and longitude, but it has not been assigned any grid coordinates. To ensure that the unit has the correct grid assignment, the model now takes the arriving location for every unit and updates its grid coordinates prior to placing the unit in a grid.

3.11 JTLS-2025-17107 NBC Attrition Crash

The CEP crashed when evaluating casualties from exposure to chemical or nuclear agents. The model was calculating the number of replacement personnel that should be killed based on the weight of excess supply category personnel, divided by the weight of a crew member. In this database, the Crew combat system had neither a resupply category nor a weight, which is legal, but caused a divide by zero crash.

Instead of using the weight of a crew member, the code was modified to use the mean weight of Combat Systems derived from the supply category for personnel. This could still be zero, so a check to avoid the divide by zero was added.

3.12 JTLS-2025-17120 User Ordered Air Mission Speeds

A user can change the speed of any air mission using one of three orders: Change Mission Parameter, Magic Air Ops, and the Change Speed quick order.

If the user does not change a mission's speed, the model will automatically tell the mission to fly at its database-designated Cruise Speed. If the mission has not been given a specific speed by a user, the model is able to adjust the mission's speed algorithmically under the following circumstances:

- When the mission is intercepting an enemy air mission.
- When the mission is a part of an Air Mission Package and is late for its designated time-on-target.

The model was not properly distinguishing between missions that were and were not given a specific speed by a player. Furthermore, once the user gave a mission a specific speed, even if that speed was to use the aircraft's Cruise Speed, the model came to the conclusion that the automatic rules could not be invoked.

Every air mission has an attribute called AM ORDERED SPEED. When this was introduced into JTLS-GO, it was only supposed to be set when a Player specifically altered the mission's speed by using one of the speed adjustment orders. Later, the attribute was being used to hold the Cruise Speed when a mission took off.

The problems were fixed, but it is important to fully understand the exact logic that is used:

- The mission has an ordered speed of zero when it is initialized and it will remain as zero unless the user enters one of the three specific change speed orders.
- When the AM ORDERED SPEED is zero, the model will:

  Adjust the mission's speed during an interception task to get into position as quickly as possible.

Adjust the mission's speed when in an Air Mission Package to ensure all missions hit their assigned targets at exactly the same time.

- When a user gives the mission specific speed instructions, the model relinquishes all responsibility for the speed of the mission. The mission will fly at that speed, even if it lands for fuel or to pickup assets, and then takes off again. This was the logic that has always been used.

- If the user has given a mission a specific speed, the orders to use a specific assigned speed can now be vacated by ordering the mission to fly at a speed of zero. When this order is processed, the mission's AM ORDERED SPEED attribute is set to zero, and its current speed is set to the aircraft's Cruise Speed. The automatic speed adjustment algorithms will be invoked when and if needed.

3.13 JTLS-2025-17122 Auto Intercept Adjust Speed

The new logic that automatically adjusts mission speed while intercepting did not function correctly in every situation. The logic was setting the mission's ordered speed attribute each time speed was adjusted. By design, the ordered speed is intended to hold the Player-specified speed, as clarified in STR JTLS-2025-17119. As a result, the original ordered speed was replaced with the adjusted interceptor speed, which prevented the mission from following the Player's original directive during and after the intercept.

The new logic was modified to treat the ordered speed as it was intended. A positive ordered speed indicates the Player has specified a speed. When the mission intercepts a victim air mission, the speed will not automatically change. An ordered speed of zero indicates the Player has not specified a speed, which permits the speed to be automatically adjusted during an intercept according to the following rules:

- If the interceptor is within optimum firing range (but not the permitted Rules of Engagement distance), between the interceptor and victim, fly at maximum speed.

- If the interceptor is within optimum firing range and ROE distance, match the victim’s speed.

- If the interceptor is not within optimum firing range and the interceptor is unarmed, fly at maximum speed.

- If the interceptor is not within optimum firing range, but within maximum range and the interceptor is armed, match the victim’s speed.

- If the interceptor is not within optimum firing range and outside of maximum range, fly at maximum speed.

3.14 JTLS-2025-17148 Protection Code For Convoy Unit Lift Capability

In a previous exercise, the model crashed when attempting to determine if a convoy could offload a type of supply that belonged to its lifted unit. The code assumed that, because this is an offload task, the convoy assets obviously had a capability to do so; otherwise, the unit load would have been canceled. It is possible that the Controller changed the data while the convoy is active, so protection code was added to check for this situation. This code change never made it into the formal JTLS-GO code base.

The situation was discovered when a check of all code changes from the previous exercise was conducted against the code base being used for this year's exercise. The protection code was reimplemented prior to exercise start and has now been properly integrated in the JTLS-GO code base.

3.15 JTLS-2025-17151 HRU Status Remained Ambush After Task Canceled

An HRU was executing an Ambush order. The Player decided to cancel the Ambush task, which correctly deleted the task from the HRU Task IMT. However, the HRU status still displayed “Ambush” in the WHIP Sitrep window. The HRU had no other pending execution tasks.

Although the routine correctly deleted the currently executing Ambush task, it failed to change the HRU's posture and mission from Ambush to Defend. Calls to the two routines that update HRU posture and mission were missing for almost every HRU task that could be canceled. If the HRU had a next task to execute, this situation was hidden because the posture immediately changed to the appropriate mission and posture.

The missing calls were added for each canceled task to change the HRU posture and mission to Defend. However, if the HRU had a Withdrawal route specified to follow after the canceled task, the two update routines were not added because the mission and posture are immediately changed to Withdraw when the Withdrawal task was executed next.

Note: If there is a pending task after the canceled task, existing logic will change the mission and posture as necessary. If no task is pending, the mission and posture will now remain in Defend, instead of the previously canceled task.

3.16 JTLS-2025-17155 Combat System Cannibalism

A Unit appeared to be losing Combat Systems for no apparent reason. The supply category for issue and resupply of the Combat System was not set to AS_USED. Because it was not AS_USED, and there were periodic usage values in the TUP, the Combat System appeared to be eating itself.

A new SVP Warning 1163 was added to identify supply categories used for Combat System issue and resupply which do not have a consumption type of AS_USED. Warning 1207 was also modified to include all periodic usage values for TUPs and SUPs.

3.17 JTLS-2025-17159 Multiple Notify External Program Events

The NATO Interactive Command and Control (ICC) system and the US Theater Battle Management Core System (TBMCS) only want to be notified of a mission delay when the mission misses its needed launch time. Delays should not be reported until a launch time is missed. On the other hand, within the model, the delay is reported as soon as the shortfall is determined. This provides the player with enough time to solve the problem. JTLS-GO supports these two requirements by scheduling a "Notify External Program" event to execute and notify ICC/TBMCS when an air mission is delayed beyond its expected launch time.

There should be only one such event scheduled for a given mission.

Investigation revealed different issues involving the notify event processing.

- A mission can go through the mission resource allocation process several times. For example, the mission can repeat the resource allocation process each time supplies arrive at the mission's home base. Each time the mission determined there were not enough resources to become launch capable, a Notify External Program event was scheduled. A call to a new subroutine was added to cancel the existing notify event before scheduling the new notify event.

- If the mission was determined to be launch capable after having been in a Delayed status, the preexisting notify event should also be canceled. A call to the new subroutine was added to cancel the existing notify event under this circumstance.

- In a related situation, when the Time on Target for a Delayed mission was changed by the Player to a later time, the existing notify event was not rescheduled as a result of the corresponding launch time being adjusted by the model. Code was added to reschedule the existing notify event.

- Finally, the routine that cancels only one existing notify event was modified to cancel all notify events for a given air mission. Given the other changes made for this STR, there should only be one notify event for a mission, The decision was to check for multiple notify events and generate a logic error if more than one such event was found and canceled.

3.18 JTLS-2025-17188 Refueled Mission Returns To Orbit After End Time

An orbiting air mission required fuel to continue its tasking. The logic selected a tanker that was some distance away. By the time the mission refueled and returned to its orbit location, the offstation time had passed. The mission then headed home later than originally scheduled. In this situation, that tanker should not have been selected to refuel the mission.

The logic that determines the feasibility of a tanker to refuel a mission was expanded to consider the off-station time. A new criterion was added to exclude a tanker (or airbase) if the flight time to or from the tanker would cause the mission to return to orbit after its orbit end time. If there is no

other feasible refueling source available, the mission simply heads home due to low fuel remaining, as usual.

3.19 JTLS-2025-17201 Mission Selected Tanker Beyond Control Range

An air mission, consisting of an aircraft type that had a control distance restriction, attempted to refuel from a tanker that was beyond the mission's maximum control distance. The mission moved toward the tanker and stopped upon reaching the maximum control distance. In this situation, that tanker should not have been chosen by the refueling logic.

New logic was added to reject refuelers (tankers or airbases) that are beyond the aircraft type's maximum control distance.

3.20 JTLS-2025-17207 Crash Change Tasks For Delayed Mission

An air mission was in Load Delay and a user entered an order to cancel one of its tasks. The last thing the code does is check if any of the changes made by the user would require a change into the scheduled launch time of the mission. This procedure crashed the model, because the mission was in a Delay State.

The logic first asked whether the mission had already started. It had not, because it could not launch due to a capability problem or an asset unavailability problem.

The logic then recomputed a new launch time. Because the mission was delayed, this logic sequence could not complete. There is no launch event scheduled for a delay mission, because the model has no idea when the delay situation will be solved.

To correct this error, an additional check was added. If the mission is in a delay state, then the entire code sequence used to recompute a new launch time is skipped. When the delay problem is solved, the model will automatically recomputes a new launch time.

3.21 JTLS-2025-17208 Cancel Mobility Mission Returns Doubled Supplies

An Air Transport Mobility mission carrying supplies landed at a destination airbase and began to offload. Before the mission had completed the offload task, a Player canceled the mission. When the mission returned home, double the supply amount was restocked at the home location.

When an Air Transport Mobility mission offloads supplies, the Offload task holds the list of supplies to remove from the mission aircraft. The estimated time to offload the supplies is calculated and a task completion event is scheduled for each aircraft in the mission. As each task completion event occurs, the supplies are removed from the mission and the task offload list is decremented accordingly.

If the mission (or the offload task itself) was canceled by the Player, the logic was incorrectly adding the remaining items in the offload list back to the mission, which caused a doubling of the

amount on board. This incorrect subroutine call was removed. As a result, the loaded supplies remain on the mission and any offloaded supplies remain at the receiving unit as expected.

3.22 JTLS-2025-17210 Fixed-Wing Aircraft Zero Landing Length Crash

JTLS-GO currently assumes that a fixed-wing aircraft requires some type of runway and must have a positive runway takeoff and landing length specified in the database. If a fixed-wing aircraft needs to "crash land" on a runway that is not longer than its landing length, and the aircraft's landing length is zero, the model will crash.

The model now checks whether the user has somehow made a fixed-wing aircraft have a zero landing length. If this is true, a Logic Error is generated, and the model indicates that no aircraft were lost when the mission crash landed.

An SVP warning was added to ensure that all fixed-wing aircraft have a landing and takeoff length greater than zero.

Several holes were also discovered in the Controller's ability to change data during execution. Code was improved to catch the following situations:

- If the Controller changes an aircraft class from Rotary to Fixed Wing, the runway lengths could end up as zero. If the user attempts to change an aircraft class to Fixed Wing when the landing length is zero, the model will accept the order but the message will inform the user that the change is illegal and internally update the AC.LAND.LENGTH to 1 ft.

- If the Controller sets the landing length to zero for a fixed-wing aircraft. The Set Aircraft Data order has been updated to refuse the order if the user attempts to change the landing length to zero if the aircraft class if Fixed Wing.

3.23 JTLS-2025-17213 Make Unit Its Own Higher Headquarters

A user submitted a Move Order and indicated that after the move, the unit should report to a new Higher HQ. The order indicated that the moving unit should become its own new Higher HQ. This caused the model and the XML Message Service (XMS) to go into an infinite loop.

The model correctly ensures that a command chain loop will not be created when changing a unit's Higher HQ. The logic did not check for the concept that the user would attempt to change its Higher HQ to itself. The code now makes this check.

3.24 JTLS-2025-17214 Random Number Seeds For ELS Mismatch

Six random number seeds were used exclusively by the Entity Level Server (ELS). They were not used in the main functions of the Scenario Initialization Program (SIP) nor by the Combat Events Program (CEP).

The Names of the seeds, and the initial values for them, were stored in the Database Development System (DDS). The names held by the DDS did not match the named expected by the SIP and CEP. This mismatch causes the programs to crash while starting.

In several places in the SIP and CEP code, the names, indexes, and values of the random number seeds were changed to be consistent with the DDS, which prevents the SIP and CEP from crashing.

3.25 JTLS-2025-17216 Manage ACM Sets Altitude To -1

After submitting a Modify ACM Shape order, an ACM was given a Default Altitude of -1.

The Default Altitude parameter does not apply to the ACM Track Shape. The code ensured that a Modify ACM order did not change the Default Altitude of an ACM Track Shape, but it failed to ask whether the user had even entered a value to modify the Default Altitude.

The Modify ACM order initializes all order parameters to negative 1. Because the code failed to ask if the user had specified a new Default Altitude, it was possible to set the Default Altitude of a non-track ACM to negative 1. The code now only changes the Default Altitude of an ACM shape if the user specified a change to the ACM attribute.

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 6.2.10.0.

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 The 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 6.2.

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 6.2 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 exist without any Combat Systems or personnel. This situation needs to be thoroughly and properly handled.

4.7 Squadron Changing Side Does Not Handle Alert Missions Properly

If a squadron is told to change sides, the model can leave alert missions at a foreign base stuck and without tasks. These types of missions must be properly handled.

4.8 CEP Incorrectly Sends Updates For Combat System IMT

The CEP properly sends out Creates for Combat System Equipment Item objects (such as trucks or artillery) when the model is started, or when a Unit was given new Combat Systems through Player action. However. the CEP then incorrectly sends out updates for Equipment Items to all Sides. This is inconsistent and the inconsistency needs to be corrected.

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
JSYMS	Java Symbols application
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
PKI	Public Key Infrastructure
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
SDC	Scenario Data Client

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 6.2.0.0 DATABASE CHANGES

Refer to the JTLS-GO 6.2.0.0 Version Description Document (VDD) for the list of database changes between the JTLS-GO 6.1 series and the JTLS-GO 6.2 series.

APPENDIX C. VERSION 6.2.0.0 REPOSITORY CHANGES

The JTLS-GO Database Team is no longer updating data in the JTLS-GO 6.2 Data Repository. All improvements will only be made in the JTLS-GO 6.3 repository.