JTLS-2018-13772 Allow Various Country Codes For C4I Systems

Richard J Kalinyak, Ellen F Roland

1.0 Summary of Model Change Request

Currently JTLS assumes that all C4I country codes are two characters long. Some systems insist on three-character country codes. Furthermore the country codes for NATO systems are different than those used by US systems. The JTLS Operational Interfaces need to support these differences.

2.0 Design Summary

This design considers several situations:

- Some C4I systems require 3 character country codes, while some systems require 2-character country codes. When one system is communicating with a different system, it is a C4I system issue and not a JTLS issue to bridge the communication barrier. On the other hand, if two independent JTLS Operational Interfaces (JOIs) are feeding two different systems, it is the responsibility of JTLS to output the proper code recognized by the system.

- Different countries have different C4I systems with different requirements with respect to the allowable country codes. It is the responsibility of the delivered JOIs to output the proper country codes for the different country requirements.

- Some exercises use fictitious countries and the fictitious country code requirements for the various systems to which JTLS can link are different.

- Not only do the various JOI processes feed real-world C4I systems, but several messages directly generated within the model are passed to C4I systems. This means that a method to alter their country codes is also needed.

Finally, this ECP is scheduled to be implemented in JTLS 5.1.0.0, but also in JTLS 5.0.12.0; therefore a methodology was developed to limit the impact on the JTLS database structure format.

3.0 Detailed Design

3.1 Background

Currently the JTLS database includes a table called “political_country”, abbreviated as POC. This table consists of three attributes:

- A 15-character Country Name (POC.NAME)
- A 2-character Country Code (POC.COUNTRY.CODE)
- The 3-digit Distributed Interactive Code (DIS) that is used to create DIS codes when interacting with entity-level simulations. (POC.DIS.CODE)

The default data delivered with JTLS currently includes 47 countries, but there are 250 countries registered with the International Standards Organization. In addition, NATO C4I systems are prepared to accept twelve different fictitious countries and internal non-country organizations such as SACLANT which can own and are responsible for NATO cross-country forces.

3.2 Static Country Code Data

The design team has researched various country code systems and we have establish the standards listed in Table 1.

Table 1. Country Code Standards

XML Tag Name	Standard Name
NATO-A2	NATO Digraph (2 letter)
NATO-A3	NATO Trigraph (3 letter)
ISO-3166-1-Numeric	International Standards Organization 3166-1 numeric
ISO-3166-1-A2	International Standards Organization 3166-1 digraph
ISO-3166-1-A3	International Standards Organization 3166-1 Trigraph
UNDP	UNDP = United Nations Development Program
WMO	WMO = World Meteorological Organization
IOC	IOC = International Olympic Committee
FIPS	FIPS = Federal Information Processing Standards
LOC	LOC = Library of Congress

JTLS will be delivered with a new static data Extended Markup Language (xml) file. This file will be located in the $JGAME/data directory and will be called Country_Codes.xml. Figure 1 shows a portion of the easy to understand format for this file.

If any user comes up with a different standard, it can easily be added to the Country_Code.xml file and no database format change will be required.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Country-Codes>
  <Country Name="Afghanistan"
           NATO-TRIGRAPH="AFG" NATO-DIGRAPH="AF" ISO-3166-1-NUMERIC="004" ISO-3166-1-TRIGRAPH="AFG"
           ISO-3166-1-DIGRAPH="AF" UNDP="AFG" WMO="AH" IOC="AFG" FIPS="AF" LOC="AF"/>
  <Country Name="Aland_Islands"
           NATO-TRIGRAPH="---" NATO-DIGRAPH="AX" ISO-3166-1-NUMERIC="248" ISO-3166-1-TRIGRAPH="ALA"
           ISO-3166-1-DIGRAPH="AX" UNDP="---" WMO="--" IOC="---" FIPS="--" LOC="--"/>
  <Country Name="Albania"
           NATO-TRIGRAPH="ALB" NATO-DIGRAPH="AL" ISO-3166-1-NUMERIC="008" ISO-3166-1-TRIGRAPH="ALB"
           ISO-3166-1-DIGRAPH="AL" UNDP="ALB" WMO="AB" IOC="ALB" FIPS="AL" LOC="AA"/>
</Country-Codes>
```

Figure 1. Example Country_Codes.xml File

3.3 Need JTLS Database Changes

Only one database change is required to implement this design. The political_country table will not change, but the user needs to indicate which country code standard was used to build the political_country table. This data parameter will be located in the single record of the global_values table. The attribute will be called Country_Code_Standard. The available options are the options listed in Table 1 and can be assigned based on a drop-down list of available standards.

3.4 JTLS Object Data Authority (JODA) Changes

The only change that is needed in the JODA Data System Protocol (JDSP) is the maximum length of the country codes that is being passed into the JODA data. It needs to be set to 3 to support each of the standards listed in the Country_Codes.xml file.

3.5 JTLS Operational Interface (JOI) Changes

The following use case better explains how this all fits together and the changes that must be made to each of the JOI processes that require Country Codes.

- The user organization builds its scenario database using whatever country code standard desired. Again, the list of standards is listed in Table 1. This means that the political_country table has an appropriate country code listed for the selected standard.

The SVP will check that this is true. For example, assume that the database has been built using the FIPS standard. If the database country code for ECUADOR political country is listed as “EQ”, an error will be generated because there is no corresponding FIPS code of EQ. The proper FIPS code for Ecuador is “EC”. This will be listed as an Error 105.

Note that this check is not fool-proof. For example, assume again that the database is built using the FIPS standard and the user has specified that the country CHINA has a country code of “CN”. This is not correct. The FIPS country code for China is “CH”, but there is a FIPS country code for “CN”, namely Comoros. The SVP cannot catch this error, and it will need to be caught during integration testing.

- Once the database is built, the model will continue to operate as it does now. The game will start and the database country code for Political_Country objects, Faction objects, Target objects, and Unit objects will be passed to the JODA. No JODA changes are required for this implementation.

- When a JOI is started it needs two pieces of data. The Country Code Standard that was used to populate the JODA and the Country Code Standard that it should output. Each time the JOI passes Country Code information to a C4I system, it must call the Country_Code conversion function. This function needs three pieces of information:

  a. Country Code from the JODA

  b. Country Code Standard used to fill the JODA. The JOI will receive this information from its interface file created by the Scenario Initialization Program and/or the Combat Events Program (CEP).

  c. Desired Country Code Standard. The JOI will receive this information from its configuration file. The data will need to be entered by Technical Control in the Interface Control Program (ICP)

  Using these three data elements, the conversion function will return the Country Code that should be sent to the C4I system.

3.6 Handling Messages Generated By The Model

As mentioned, a JOI is not the only process within JTLS that will need to alter the database Country Code for interface to C4I systems. The following methods are also available to pass information to C4I systems:

a. It is possible for a WHIP to e-mail a C4I system.

b. The Message Delivery Program (MDP) can e-mail or pass via a direct socket link messages to a C4I system.

Each of these methods use the same process to render messages. The library that renders messages will also have access to the Country_Code conversion function discussed in Section 3.5. The only issue is to alter the appropriate Message Definition Files (MDF) to indicate when the conversion function should be called.

To understand how this new Country_Code conversion function will work, it is best to relate it to the similarly implemented message rendering Location function Currently, everywhere that a location is output in a message, the MDF calls the LOCATION function to place the location in the formatted message style-sheet. As part of the WHIP's Message Browser and the MDP Graphical User Interface (GUI), the user chooses the desired location format, either Latitude/Longitude or Military Grid System (MGS),

This means that everywhere that the model outputs the database value POC.COUNTRY.CODE, within a message, the MDF must be changed to call the conversion function, COUNTRY_CODE(DataAtomID). As part of the rendering process, the user must specify the desired Country Code format that should be displayed within the rendered message. The user will have all of the formats listed in Table 1. from which to choose. Each time the user selects a new format, the message being viewed will be re-rendered and the new country code format will be displayed.

3.7 Data Repository

The Design Team believes that no changes are required to the current 47 record political_country table. If users need to add any countries the database builder can do so and enter the code from the $JGAME/country_code.xml file for desired standard which the data is built.

3.8 Implementation Differences Between JTLS 5.1.0.0 and JTLS 5.0.12.0

It is not possible to change the database format for the JTLS 5.0 series of releases. This means that the new database parameter called country_code_standard cannot exist in the database. Within JTLS 5.0.12.0, the JOI which need to convert the country code will need to have the user specify the input standard on the user interface. It will continue to get the desired output standard from the ICP generated configuration file.

The message rendering capability, discussed in Section 3.6 will not be implemented in JTLS 5.0.12.0.

4.0 Data Changes

COUNTRY.CODE.STANDARD

- Dimension: Variable
- Mode: Text

- Unit of Measure: Must be one of the support Country Code specifications listed in the file $JGAME/data/Country_Code.xml.
- Range: N/A
- Definition: This is the Country Code standard that was used to fill the variable POC COUNTRY CODE withing the Political Country table.
- Relationships: This parameter is related to the Country Codes (POC COUNTRY CODE) entered for each of the Political Country records. The file $JGAME/data/Country_Code.xml lists the legal country codes that can be entered for the selected standard.

5.0 Order Changes

No order changes are being made to support this ECP. This means that the database parameter COUNTRY CODE STANDARD cannot be changed after game start.

6.0 JODA Changes

No JODA changes are required to support this ECP.

7.0 Test Plan

To fully test this ECP, two real-world C4I systems are needed, in which different country code standards are required, one that requires the standard to which the database was built and one that requires a different standard. The test plan as written assumes that this is not possible and has developed the tests so you build two databases, one that use the Country Code Standard that is needed by the C4I system, and one that uses an alternative standard.

If two different systems are available, do not execute the steps that indicate the database needs to be changed. This steps are marked in Red.

7.1 Test New SVP Error

Purpose: The purpose of this test is to check that the new SVP Error check for allowable country codes works as designed.

Step 1: For a given database, note the name of the COUNTRY CODE STANDARD that was used to create the database.

Step 2: Select one of the Political Country records and change its assigned Country Code to a non-existent country code. Make sure the change country code does not exist in the Country_Code.xml file held in the $JGAME/data directory.

Step 3: Download the database, and run the SVP.

Expected Results: Error 105 should be generated.

Step 4: Select each of the correction options for Error 105 in the DDSC.

Expected Results: Each correction option should execute as expected.

Step 5: Correct the Error 105 generated.

Step 6: Change the database parameter COUNTRY CODE STANDARD

Step 7: Download the database and run the SVP

Expected Results: Several of the Political Country records should report an Error 105.

Step 8: Correct the database to get rid of all of the Error 105 records

Expected Results: All should work as expected.

7.2 Test OTH-Gold JOI Output

Purpose: The purpose of this test is to insure that the OTH-Gold Message Service outputs the proper country code required by the C4I system.

Step 1: Select a database that is created with the Country Code Standard needed by the primary OTH-Gold C4I device.

Step 2: Bring up the game, and indicate in the ICP that the same Country Code Standard should be output.

Step 3: Start the OTH-Gold JOI and have it connect to the C4I system.

Expected Results: All of the messages should be accepted by the C4I system.

Step 4: Change the database and have it use a different Country Code Standard. Alternatively connect to a OTH-Gold System that requires a country code other than the standard used to build the existing database.

Step 5: Indicate in the ICP that this JOI should output a standard other than the standard used to build the database.

Step 6: Start the OTH-Gold JOI and have it connect to the C4I system.

Expected Results: All of the messages should be accepted by the C4I system.

7.3 Test TACELINT JOI Output

Purpose: The purpose of this test is to insure that the TACELINT Message Service outputs the proper country code required by the C4I system. Normally these C4I systems are not available for testing. Each of the tests assume that you are outputting the TACELINT messages to a terminal screen and viewing the message for the proper country code.

Step 1: Since this test does not expect to link to a specific C4I system, select any available 5.1 database

Step 2: Create two TACELINT message services in the ICP. Have one indicate that should output the Country Code Standard that was used to build the selected database and should feed a socket associated with test jtls_server one. The other TACELINT message service should indicate that it should output a different Country Code Standard and should feed a socket associated with a second test jtls_server.

Step 3: In separate terminal windows, start to two test jtls_server processes.

Step 4: Run the game and the two TACELINT Message Services.

Expected Results: The messages delivered to the first jtls_server should have the country codes that are in the database. The messages delivered to the second jtls_server should have the country codes as specified for the second selected Country Code Standard.

7.4 Test Transactional Operational Interface JTOI) Output

Purpose: The purpose of this test is to insure that the various JTOI processes work with the new country code standard. These processes include the connection to the Theater Battle Management Core system (TBMCS), the Integrated Command and Control (ICC) system and the Norther European Command and Control Information System (NECCIS).

Step 1: Bring up the game,

Step 2: Start the TBMCS JTOI and indicate on the startup panel that it should output the country code standard with which the database was built.

Step 3: Have it connect to TBMCS.

Expected Results: All of the messages should be accepted by TBMCS.

Step 4: Change the database and have it use a different Country Code Standard, and restart the game,

Step 5: Indicate on the JTOI Startup panel that it should output a standard other than the standard used to build the database.

Step 6: Start the TBMCS JTOI and have it connect to the C4I system.

Expected Results: All of the messages should be accepted by the C4I system.

Repeat the same six steps for ICC and NECCIS.

7.5 Test Message Browser Output

Purpose: The purpose of this test is to insure that the WHIP Message Browser allows the user to change the Country Code standard that should be used to render messages

Step 1: Using any database, generate a NATO Enemy SITREP report.

Step 2: Sequentially go through each of the Country Code Standards and view the message with each standard.

Expected Results: For the enemy units being reported, the message should show the appropriate country code given the selected Country Code Standard.

7.6 Test Message Delivery Program Output.

Purpose: The purpose of this test is to insure that the Message Delivery Program (MDP) allows the user to change the Country Code standard that should be used to render messages

Step 1: Setup the MDP to have two different message profiles. One that uses a two-character Country Code and one that uses a three-character country code. Have each message profile use a different delivery profile to put the messages in two different directories on the server.

Step 2: Generate several NATO Enemy SITREP (ENESITREP) report.

Expected Results: The messages should be placed in the two different directories. One directory should have the messages rendered with the two-character country codes and the other directory should have the messages using the three-character country codes.

Step 3: Save the MDF configuration and exit out of the MDP GUI.

Step 4: Restart the MDP GUI and view the defined profiles.

Expected Results: The definition of the profiles should be exactly as you had them before exiting out of the MDP GUI.

Step 5: Change one of the profiles to display Location using the Military Grid System (MGRS).

Step 6: Save the profile definition.

Step 7: Generate some new ENESITREP.

Expected Results: The newly generated ENESITREP should still show the properly selected Country Codes and the location in MGRS.

7.7 Test Changing Country Code During Game Execution

Purpose: The purpose of this test is to insure that the Controller can change the Country Code assigned to a Political Country during game execution.

Step 1: Bring up a game and note the Country Code Standard that is being used.

Step 2: Bring up the SET POLITICAL COUNTRY Order and select an Political Country that is referenced by one of the units that is being reported in your ENESITREP Reports from the previous tests.

Step 3: Change the Country Code for that Political Country. Note it does not need to be the correct Country Code, but it must exist in the Country_Code.xml file for the Country Code Standard being used.

Step 4: Generate another NATO Enemy Situation Report and view in the Message Browser.

Expected Results: The new Country Code should be displayed in the message.

Step 5: Set the Message Browser so it uses a different Country Code Standard. View the same ENESITREP.

Expected Results: The country code that is the equivalent to the new country code that you entered should be displayed.

7.8 Test Reject Country Code Change During Game Execution

Purpose: the purpose of this test is to insure that the Controller cannot incorrectly change the Country Code assigned to a Political Country.

Step 1: Bring up a game and note the Country Code Standard that is being used.

Step 2: Bring up the SET POLITICAL COUNTRY Order and select an Political Country that is referenced by one of the units that is being reported in your ENESITREP Reports from the previous tests.

Step 3: Change the Country Code for that Political Country to a code that does not exist for the database Country Code Standard.

Expected Results: The SET POLITICAL COUNTRY Order should be sent to the model, but the model should reject the order and send an appropriate rejection message to the Controller's WHIP.