T308: Acceptance Testing for Advanced Transportation Controller Application Programming Interface Software

Table of Contents

1. Module Description ................................................................. 2
2. Introduction/Purpose ............................................................. 2
3. Samples/Examples ................................................................ 3
4. Reference to Other Standards ............................................. 10
5. Glossary ............................................................................. 11
6. References ........................................................................... 13
7. Study Questions .................................................................... 14
8. Icon Guide ........................................................................... 15
1. Module Description

Module T308: Acceptance Test for Advanced Transportation Controller Application Programming Interface Software provides training for the testing API Software. The API Software is to conform to the ATC 5402 v02 Standard.

The following are recommended prerequisites in the ATC curriculum for students taking this course.

- A207b: Building an ITS infrastructure Based on the ATC 5201 Standard Part 2 of 2
- A208: Using the ATC 5401 API Standard to Leverage ITS Infrastructure
- A307a: Understanding User Needs for Advanced Transportation Controller (ATC) Based on ATC 5201 Standard v06
- A307b: Understanding Requirements for Advanced Transportation Controller (ATC) Based on ATC 5201 Standard v06
- T307: Applying Your Test Plan to the Advanced Traffic Controller based on ATC 5201 Standard v06

The following are recommended prerequisites in the testing curriculum for students taking this course.

- T101: Introduction to ITS Standards Testing
- T201: How to Write a Test Plan
- T202: Overview of Test Design Specifications, Test Case Specifications, and Test Procedures
- T203: Part 1 of 2: How to Develop Test Cases for an ITS Standards-Based Test Plan, Part 1 of 2
- T203: Part 2 of 2: How to Develop Test Cases for an ITS Standards-Based Test Plan, Part 2 of 2
- T204: Part 1 of 2: How to Develop a Test Procedure for ITS Standards-based Test Plan
- T204: Part 2 of 2: How to Develop a Test Procedure for ITS Standards-based Test Plan

2. Introduction/Purpose

The Advanced Transportation Controller (ATC) family of standards provides an open architecture hardware and software platform that can support a wide variety of Intelligent Transportation Systems (ITS) field applications. These standards are characterized by their modularity, support of multiple and concurrent application programs, and design to facilitate the adoption of new technologies.

The ATC 5401 Application Programming Interface (API) Standard v02 defines API Software that provides both user and programmatic interfaces for transportation controller units that conform to the ATC 5201 Transportation Controller Standard (ATC units). The API Software allows application programs to be written so that they may run on any ATC unit regardless of the manufacturer. It also allows multiple application programs to be compatible on a single ATC unit by sharing the fixed resources of the controller.

As part of the API Reference Implementation (APIRI) Project, the API Validation Suite (APIVS) software was developed that may be used to test API Software resident on an ATC unit. The APIVS Software is maintained in an open source software (OSS) environment and is available to anyone. Test documentation based on IEEE 829-2008 that leverages that APIVS Software is also provided.
Module T308 focuses on testing the API Software based on the ATC S401 Standard. At the conclusion of this module, participants will be able to: understand the purpose of the APIVS Software; use test documentation to specify API Software acceptance testing; use the APIVS Software to perform API Software testing; and interpret and report the results of testing.

3. Samples/Examples

From the ATC APIRI Test Plan v01.05, Section 3, Features to be Tested, Table 1.

<table>
<thead>
<tr>
<th>Test ID</th>
<th>Document Name</th>
<th>Brief Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APIRI.TDS.2001</td>
<td>APIRI Test Design Specification 1</td>
<td>Test All APIRI FPUI Required Features</td>
</tr>
<tr>
<td>APIRI.TDS.3001</td>
<td>APIRI Test Design Specification 2</td>
<td>Test All APIRI FIO Required Features</td>
</tr>
<tr>
<td>APIRI.TDS.4001</td>
<td>APIRI Test Design Specification 3</td>
<td>Test All APIRI TOD Required Features</td>
</tr>
<tr>
<td>APIRI.TCS.2010</td>
<td>APIRI Test Case Specification 1</td>
<td>FPUI Text UI Virtual Displays</td>
</tr>
<tr>
<td>APIRI.TCS.2020</td>
<td>APIRI Test Case Specification 2</td>
<td>FPUI Front Panel Manager</td>
</tr>
<tr>
<td>APIRI.TCS.2030</td>
<td>APIRI Test Case Specification 3</td>
<td>FPUI Character Set and Screen Attributes</td>
</tr>
<tr>
<td>APIRI.TCS.2040</td>
<td>APIRI Test Case Specification 4</td>
<td>FPUI Reading and Writing Data</td>
</tr>
<tr>
<td>APIRI.TCS.2050</td>
<td>APIRI Test Case Specification 5</td>
<td>FPUI Special Characters</td>
</tr>
<tr>
<td>APIRI.TCS.2070</td>
<td>APIRI Test Case Specification 6</td>
<td>FPUI Key Mapping</td>
</tr>
<tr>
<td>APIRI.TCS.2080</td>
<td>APIRI Test Case Specification 7</td>
<td>FPUI Asynchronous Notification and Focus</td>
</tr>
<tr>
<td>APIRI.TCS.2090</td>
<td>APIRI Test Case Specification 8</td>
<td>FPUI Raw Data Handling</td>
</tr>
<tr>
<td>APIRI.TCS.2100</td>
<td>APIRI Test Case Specification 9</td>
<td>API Version Information (All Libraries)</td>
</tr>
<tr>
<td>APIRI.TCS.3010</td>
<td>APIRI Test Case Specification 10</td>
<td>General FIO Operations</td>
</tr>
<tr>
<td>APIRI.TCS.3020</td>
<td>APIRI Test Case Specification 11</td>
<td>FIO Inputs and Outputs</td>
</tr>
<tr>
<td>APIRI.TCS.3030</td>
<td>APIRI Test Case Specification 12</td>
<td>FIO Channel Mapping</td>
</tr>
<tr>
<td>APIRI.TCS.3040</td>
<td>APIRI Test Case Specification 13</td>
<td>FIO Filtered Inputs and Transition Buffering</td>
</tr>
<tr>
<td>APIRI.TCS.3050</td>
<td>APIRI Test Case Specification 14</td>
<td>FIO Frame Frequency</td>
</tr>
<tr>
<td>APIRI.TCS.3060</td>
<td>APIRI Test Case Specification 15</td>
<td>FIO Failed State and Fault Monitoring</td>
</tr>
<tr>
<td>APIRI.TCS.3070</td>
<td>APIRI Test Case Specification 16</td>
<td>FIO Watchdog Outputs</td>
</tr>
<tr>
<td>APIRI.TCS.3080</td>
<td>APIRI Test Case Specification 17</td>
<td>FIO Device Status</td>
</tr>
<tr>
<td>APIRI.TCS.3090</td>
<td>APIRI Test Case Specification 18</td>
<td>FIO Health Monitor</td>
</tr>
<tr>
<td>APIRI.TCS.3100</td>
<td>APIRI Test Case Specification 19</td>
<td>FIO CMU Configuration</td>
</tr>
<tr>
<td>APIRI.TCS.3110</td>
<td>APIRI Test Case Specification 20</td>
<td>FIO Module Status</td>
</tr>
<tr>
<td>APIRI.TCS.3120</td>
<td>APIRI Test Case Specification 21</td>
<td>FIO Asynchronous Notification</td>
</tr>
<tr>
<td>APIRI.TCS.3130</td>
<td>APIRI Test Case Specification 22</td>
<td>FIO Dark Channel Mapping</td>
</tr>
<tr>
<td>APIRI.TCS.4010</td>
<td>APIRI Test Case Specification 23</td>
<td>TOD Time Handling Functions</td>
</tr>
<tr>
<td>APIRI.TCS.6010</td>
<td>APIRI Test Case Specification 24</td>
<td>FPM and ATC Configuration Menu</td>
</tr>
<tr>
<td>APIRI.TCS.6020</td>
<td>APIRI Test Case Specification 25</td>
<td>System Configuration Utilities</td>
</tr>
<tr>
<td>APIRI.TCS.6030</td>
<td>APIRI Test Case Specification 26</td>
<td>Intrinsic API Requirements</td>
</tr>
</tbody>
</table>

The runAPIVS.sh Linux Shell Script.

#!/bin/sh

# ATC 5401 API Reference Implementation Project
#
#     Filename: runAPIVS
#    File Type: Linux shell script
#    Test Case: many
#  Description: run VSE from USB at startup on specific test cases
#
# Date      Revision    Description
# 2/24/16   1.0         initial release

# start async loopback driver; add symbolic links
insmod /media/sda1/APIVS/bin/tty0tty.ko
ln -s /dev/tnt0 /dev/sp6_loopback_a
ln -s /dev/tnt1 /dev/sp6_loopback_b

# start the API Front Panel Manager (loopback mode)
modprobe front_panel
FrontPanelManager /dev/sp6_loopback_a 1>/dev/null 2>&1 &
MasterSelection 1>/dev/null 2>&1 &

# start sync loopback driver; fio driver; add symbolic links
rmmod fiodriver
insmod /media/sda1/APIVS/bin/virtual-loopback-sync.ko
insmod /media/sda1/APIVS/bin/fiodriver.ko apivs=1
ln -s /dev/vlsync0 /dev/sp3s_loopback_a
ln -s /dev/vlsync1 /dev/sp3s_loopback_b
ln -s /dev/vlsync2 /dev/sp5s_loopback_a
ln -s /dev/vlsync3 /dev/sp5s_loopback_b
ln -s /dev/vlsync4 /dev/sp8s_loopback_a
ln -s /dev/vlsync5 /dev/sp8s_loopback_b

# initialize test counters
pass_count=0
fail_count=0

# a couple of useful subroutines
clear_test_line() {
    printf \x1b[3;1f >/dev/sp6
    printf " " >/dev/sp6
    printf \x1b[3;1f >/dev/sp6
}
print_test_result() {
    if [ "$?" == "0" ]; then
        printf "PASS" >/dev/sp6
        pass_count=$((pass_count + 1))
    else
        printf "FAIL" >/dev/sp6
        fail_count=$((fail_count + 1)); fi
    printf \x1b[8;1f >/dev/sp6
    printf "Test cases passed:%d failed:%d" "$pass_count" "$fail_count" >/dev/sp6
}

# keep backlight on
printf \033[<5h >/dev/sp6
sleep 1
}
reset_modules() {
    # start the API Front Panel Manager (loopback mode)
    killall MasterSelection
    killall FrontPanelManager
    rmmod front_panel
    modprobe front_panel
    FrontPanelManager /dev/sp6_loopback_a 1>/dev/null 2>&1 &
    MasterSelection 1>/dev/null 2>&1 &
    # start fio driver
    rmmod fiodriver
    insmod /media/sda1/APIVS/bin/fiodriver.ko apivs=1
}
misc_test_C6020() {
    # Check for uClibc load object identifier (APIVS[3.8])
    LD_TRACE_LOADED_OBJECTS=1 vse | grep -q "ld-uClibc"
    if [ $? != 0 ]
        echo "$(date -u): vse is not compatible with uClibc"
    then
        echo "$(date -u): vse is compatible with uClibc"
        /tmp/C6020_log.txt
        retval=1
    else
        echo "$(date -u): vse is compatible with uClibc"
        /tmp/C6020_log.txt
        retval=0
    fi
    mv /tmp/C6020_log.txt ./
    sync
    return $retval
}
misc_test_C6030() {
    # Check for ELF format (APIR3.5.2[5])
    elfmagic="7f454c46"
FILES="/usr/lib/libfpui.so
/usr/lib/libfio.so
/usr/lib/libtod.so"

for f in $FILES
do
    if [ $(od -An -N4 -tx4 <$f) == $elfmagic ]; then
        echo "$(date -u): $f is ELF format file" >>=/tmp/C6030_log.txt
        retval=0
    else
        echo "$(date -u): $f is not ELF format file" >>=/tmp/C6030_log.txt
        retval=1
    fi
done
mv /tmp/C6030_log.txt ./
sync
return $retval

# set the conformance level this run (1,2,3)
LEVEL=3

# delete old log files? (FALSE will append if exists)
DELETE_LOGS=TRUE

# how many times to run?
loop_count=1

# loop through all specified test cases
while [ $loop_count -ge 1 ]
do
    reset_modules
    if [ "$DELETE_LOGS" == TRUE ]; then rm C1110_log.xml; fi
    clear_test_line; printf "Testing APIVS.TCS.1110... " >/dev/sp6
    vse -L $LEVEL -c ./VS_config_1.txt -i C1110_in.xml -o C1110_log.xml
    print_test_result

    reset_modules
    if [ "$DELETE_LOGS" == TRUE ]; then rm C1120_log.xml; fi
    clear_test_line; printf "Testing APIVS.TCS.1120... " >/dev/sp6
    vse -L $LEVEL -c ./VS_config_1.txt -i C1120_in.xml -o C1120_log.xml
    print_test_result

    reset_modules
    if [ "$DELETE_LOGS" == TRUE ]; then rm C1130_log.xml; fi
    clear_test_line; printf "Testing APIVS.TCS.1130... " >/dev/sp6
    vse -L $LEVEL -c ./VS_config_1.txt -i C1130_in.xml -o C1130_log.xml
    print_test_result

    reset_modules
    if [ "$DELETE_LOGS" == TRUE ]; then rm C1150_log.xml; fi
    clear_test_line; printf "Testing APIVS.TCS.1150... " >/dev/sp6
    vse -L $LEVEL -c ./VS_config_1.txt -i C1150_in.xml -o C1150_log.xml
    print_test_result

    reset_modules
    if [ "$DELETE_LOGS" == TRUE ]; then rm C1160_log.xml; fi

clear_test_line; printf "Testing APIVS.TCS.1160... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1160_in.xml -o C1160_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1310_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1310... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1310_in.xml -o C1310_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1320_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1320... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1320_in.xml -o C1320_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1330_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1330... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1330_in.xml -o C1330_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1350_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1350... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1350_in.xml -o C1350_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1410_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1410... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1410_in.xml -o C1410_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1420_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1420... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1420_in.xml -o C1420_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1430_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1430... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1430_in.xml -o C1430_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C1450_log.xml; fi
clear_test_line; printf "Testing APIVS.TCS.1450... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C1450_in.xml -o C1450_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2010_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2010... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2010_in.xml -o C2010_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2020_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2020... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2020_in.xml -o C2020_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2030_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2030... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2030_in.xml -o C2030_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2040_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2040... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2040_in.xml -o C2040_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2050_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2050... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2050_in.xml -o C2050_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2070_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2070... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2070_in.xml -o C2070_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2080_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2080... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2080_in.xml -o C2080_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2090_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2090... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2090_in.xml -o C2090_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C2100_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.2100... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C2100_in.xml -o C2100_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3010_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3010... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3010_in.xml -o C3010_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3020_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3020... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3020_in.xml -o C3020_log.xml
print_test_result
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3030_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3030... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3030_in.xml -o C3030_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3040_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3040... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3040_in.xml -o C3040_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3050_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3050... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3050_in.xml -o C3050_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3060_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3060... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3060_in.xml -o C3060_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3070_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3070... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3070_in.xml -o C3070_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3080_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3080... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3080_in.xml -o C3080_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3090_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3090... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3090_in.xml -o C3090_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3100_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3100... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3100_in.xml -o C3100_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3110_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3110... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3110_in.xml -o C3110_log.xml
print_test_result

reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3120_log.xml; fi
clear_test_line; printf "Testing APIRI.TCS.3120... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3120_in.xml -o C3120_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C3130_log.xml; fi

clear_test_line; printf "Testing APIRI.TCS.3130... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C3130_in.xml -o C3130_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C4010_log.xml; fi

clear_test_line; printf "Testing APIRI.TCS.4010... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C4010_in.xml -o C4010_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C6010_log.xml; fi

clear_test_line; printf "Testing APIRI.TCS.6010... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C6010_in.xml -o C6010_log.xml
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C6020_log.txt; fi

clear_test_line; printf "Testing APIRI.TCS.6020... " >/dev/sp6
misc_test_C6020
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C6030_log.txt; fi

clear_test_line; printf "Testing APIRI.TCS.6030... " >/dev/sp6
misc_test_C6030
print_test_result
reset_modules
if [ "$DELETE_LOGS" == TRUE ]; then rm C7030_log.xml; fi

clear_test_line; printf "Testing APIRI.TCS.7030... " >/dev/sp6
vse -L $LEVEL -c ./VS_config_1.txt -i C7030_in.xml -o C7030_log.xml
print_test_result

cmp /tmp/*log.xml ./
sync

loop_count=$((loop_count - 1))
done

4. Reference to Other Standards

http://standards.ieee.org/index.html

Institute of Transportation Engineers, *ATC 5401 Application Programming Interface (API) Standard for the Advanced Transportation Controller (ATC) v02*. ATC Joint Committee, 15 September 2013.
http://www.ite.org/standards/index.asp
Institute of Transportation Engineers, *User Comment Draft ATC 5201 Advanced Transportation Controller (ATC) Standard Version 06.10. ATC Joint Committee, 30 July 2012.*


5. Glossary

To include additional descriptions/acronyms used primarily in the module.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AASHTO</td>
<td>American Association of State Highway and Transportation Officials.</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface.</td>
</tr>
<tr>
<td>API Utilities</td>
<td>API software that is used for setting system-wide purposes on an ATC controller unit.</td>
</tr>
<tr>
<td>APIRI Project</td>
<td>Entire project managed by ATC APIRI PMP v01.01 Project Management Plan (PMP) for the Advanced Transportation Controller (ATC) Application Programming Interface (API) Reference Implementation Project including software, hardware and documentation.</td>
</tr>
<tr>
<td>APIRI Software</td>
<td>API Reference Implementation (software). API software developed as part of the ATC APIRI Project.</td>
</tr>
<tr>
<td>APIVS Software</td>
<td>API Validation Suite Software.</td>
</tr>
<tr>
<td>Application Program</td>
<td>Any program designed to perform a specific function directly for the user or, in some cases, for another application program. Examples of application programs include word processors, database programs, Web browsers and traffic control programs. Application programs use the services of a computer's O/S and other supporting programs such as an application programming interface.</td>
</tr>
<tr>
<td>ATC</td>
<td>Advanced Transportation Controller.</td>
</tr>
<tr>
<td>ATC Device Drivers</td>
<td>Low-level software not included in a typical Linux distribution that is necessary for ATC-specific devices to operate in a Linux O/S environment.</td>
</tr>
<tr>
<td>Board Support Package</td>
<td>Software usually provided by processor board manufacturers that provide a consistent software interface for the unique architecture of the board. In the case of the ATC, the Board Support Package also includes the O/S.</td>
</tr>
<tr>
<td>BOM</td>
<td>Bill of Materials. A list of the raw materials, sub-assemblies, intermediate assemblies, sub-components, parts and the quantities of each needed to manufacture an end product.</td>
</tr>
<tr>
<td>BSP</td>
<td>See Board Support Package.</td>
</tr>
<tr>
<td>ConOps</td>
<td>Concept of Operations.</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit. A programmable logic device that performs the instruction, logic and mathematical processing in a computer.</td>
</tr>
<tr>
<td>Device Driver</td>
<td>A software routine that links a peripheral device to the operating system. It acts like a translator between a device and the application programs that use it.</td>
</tr>
<tr>
<td>FIO</td>
<td>Field Input and Output.</td>
</tr>
<tr>
<td>FPUI</td>
<td>Front Panel User Interface.</td>
</tr>
<tr>
<td>H/W</td>
<td>Hardware.</td>
</tr>
<tr>
<td>I/O</td>
<td>Input/Output.</td>
</tr>
<tr>
<td>IEC</td>
<td>International Electrotechnical Commission.</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers.</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization.</td>
</tr>
<tr>
<td>ITE</td>
<td>Institute of Transportation Engineers.</td>
</tr>
<tr>
<td>ITS</td>
<td>Intelligent Transportation Systems.</td>
</tr>
<tr>
<td>JC</td>
<td>Joint Committee.</td>
</tr>
<tr>
<td>JPO</td>
<td>Joint Program Office.</td>
</tr>
<tr>
<td>Linux</td>
<td>Low-level software that is freely available in the Linux community for use with common hardware components operating in a standard fashion.</td>
</tr>
<tr>
<td>Linux Kernel</td>
<td>The Unix-like operating system kernel that was begun by Linus Torvalds in 1991. The Linux Kernel provides general O/S functionality. This includes functions for things typical in any computer system such as file I/O, serial I/O, interprocess communication and process scheduling. It also includes Linux utility functions necessary to run programs such as shell scripts and console commands. It is generally available as open source (free to the public). The Linux Kernel referenced in this standard is defined in the ATC Controller Standard Section 4.3.5, Appendix A and Appendix B.</td>
</tr>
<tr>
<td>Loopback Driver</td>
<td>A virtual device driver that loops back the output ports to a device to the input ports from a device without actually going through the physical device.</td>
</tr>
<tr>
<td>Mechanical Drawing</td>
<td>A drawing to scale of a machine, machine component, or device from which dimensions can be taken for manufacturing.</td>
</tr>
<tr>
<td>N/A</td>
<td>Not Applicable.</td>
</tr>
<tr>
<td>O/S</td>
<td>Operating System.</td>
</tr>
<tr>
<td>Operational User</td>
<td>A technician or transportation engineer who uses the controller to perform its operational tasks.</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board.</td>
</tr>
<tr>
<td>PMP</td>
<td>Project Management Plan.</td>
</tr>
<tr>
<td>RI</td>
<td>Reference Implementation.</td>
</tr>
<tr>
<td>RTC</td>
<td>Real-Time Clock.</td>
</tr>
</tbody>
</table>
### Term Definition

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>S/W</td>
<td>Software.</td>
</tr>
<tr>
<td>Schematic Diagram</td>
<td>A diagram that shows, by means of graphic symbols, the electrical connections and functions of a specific circuit arrangement.</td>
</tr>
<tr>
<td>SDD</td>
<td>Software Design Document or Software Design Description.</td>
</tr>
<tr>
<td>SDO</td>
<td>Standards Development Organization.</td>
</tr>
<tr>
<td>SE</td>
<td>Systems Engineer.</td>
</tr>
<tr>
<td>Software Validation</td>
<td>The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.</td>
</tr>
<tr>
<td>SOW</td>
<td>Statement of Work.</td>
</tr>
<tr>
<td>SRS</td>
<td>Software Requirements Specification.</td>
</tr>
<tr>
<td>TBD</td>
<td>To Be Determined.</td>
</tr>
<tr>
<td>Tester</td>
<td>A user developer, test engineer or test technician capable of operating the API Validation Suite described by this document.</td>
</tr>
<tr>
<td>TFCS</td>
<td>Transportation Field Cabinet System.</td>
</tr>
<tr>
<td>TOD</td>
<td>Time of Day.</td>
</tr>
<tr>
<td>TOPR</td>
<td>Task Order Proposal Request.</td>
</tr>
<tr>
<td>US</td>
<td>United States.</td>
</tr>
<tr>
<td>USDOT</td>
<td>United States Department of Transportation.</td>
</tr>
<tr>
<td>User Developer</td>
<td>A software developer that designs and develops programs for controllers.</td>
</tr>
<tr>
<td>Walkthrough</td>
<td>A step-by-step presentation by the author of a document in order to gather information and to establish a common understanding of its content.</td>
</tr>
<tr>
<td>WG</td>
<td>Working Group.</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language.</td>
</tr>
</tbody>
</table>

### 6. References

Application Programming Interface (API) Reference Implementation Project

API Reference Implementation (APIRI) Repository
[https://github.com/apiriadmin/APIRI](https://github.com/apiriadmin/APIRI)

API Validation Suite (APIVS) Repository
[https://github.com/apiriadmin/APIVS](https://github.com/apiriadmin/APIVS)

Notepad++. General purpose editing tool for software related files Color coding and formatting of XML text files
[http://notepad-plus-plus.org](http://notepad-plus-plus.org)

XML Differences. Online comparison of XML files.
[www.corefiling.com/opensource/xmldiff.html](http://www.corefiling.com/opensource/xmldiff.html)
XmlGrid. Online editor displays in formatted XML text or in grids (tables).

XML Viewer. Online editor displays in formatted XML text or in tree view.
www.codebeautify.org/xmlviewer

7. Study Questions
To include the quiz/poll questions and answer choices as presented in the PowerPoint slide to allow students to either follow along with the recording or refer to the quiz at a later date in the supplement.

1. What controller software is NOT traditionally tested by agencies?
   a) Data Collection Software
   b) Signal Control Software
   c) Application Programming Interface Software
   d) Ramp Meter Software

2. What document is used to specify the inputs and outputs for a particular test of the API Software?
   a) Test Design
   b) Test Procedure Specification
   c) Test Plan
   d) Test Case Specification

3. What is not an appropriate reason to edit the runAPIVS shell script?
   a) Turn off all test output
   b) Change the number of iterations on a test
   c) Change the conformance report logging
   d) Select a subset of the existing test cases

4. True or False: It’s a good idea to always log as much information as possible on all tests.
   a) True
   b) False
8. Icon Guide

The following icons are used throughout the module to visually indicate the corresponding learning concept listed out below, and/or to highlight a specific point in the training material.

1) **Background information**: General knowledge that is available elsewhere and is outside the module being presented. This will be used primarily in the beginning of slide set when reviewing information readers are expected to already know.

2) **Tools/Applications**: An industry-specific item a person would use to accomplish a specific task, and applying that tool to fit your need.

3) **Remember**: Used when referencing something already discussed in the module that is necessary to recount.

4) **Refer to Student Supplement**: Items or information that are further explained/detailed in the Student Supplement.

5) **Example**: Can be real-world (case study), hypothetical, a sample of a table, etc.

6) **Checklist**: Use to indicate a process that is being laid out sequentially.