

# Amber Tool Validation

Department Name

July 16, 2022

## Abstract

In this document Amber is the Configuration Item being validated and Report Package is the Amber Report Package. Amber is validated after this Report Package has been executed, test evidence has been obtained, and test evidence supports the conclusion Intended Use Requirements (IUR) are satisfied. This document is stored in Acme Corporation's (Company) Quality Management System after Amber has been validated.

## Contents

<b>Approval Signatures</b>	<b>3</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Overview	3
1.2 Purpose	3
1.3 Scope	3
1.4 Deviations	3
1.5 Tool Validation Objectives	4
1.6 General Terms	4
1.7 General Acronyms	5
1.8 References	5
1.9 Training	5
1.10 Tool Validation Test Approach	5
1.11 Configuration Management	5
1.12 Test Plan Instructions	5
1.13 Test Plan Storage and Review	6
<b>2 Requirements</b>	<b>7</b>
<b>3 Test Plan Overview</b>	<b>8</b>
<b>4 Test Evidence</b>	<b>9</b>
4.1 Test Plan: master	9
4.1.1 Test Suite: options	9
4.1.2 Test Case: browser	9
4.1.3 Test Case: case	11
4.1.4 Test Case: default	12
4.1.5 Test Case: environment	13
4.1.6 Test Case: file	14
4.1.7 Test Case: language	15
4.1.8 Test Case: nodryrun	19
4.1.9 Test Case: obliterate	20
4.1.10 Test Case: plan	21
4.1.11 Test Case: simulate	22
4.1.12 Test Case: suite	23
4.1.13 Test Case: verbose	24
4.1.14 Test Case: version	25

4.1.15	Test Case: writer . . . . .	26
4.1.16	Test Suite: substitute . . . . .	27
4.1.17	Test Case: browser . . . . .	27
4.1.18	Test Case: extend-path . . . . .	29
4.1.19	Test Case: home . . . . .	30
4.1.20	Test Case: language . . . . .	31
4.1.21	Test Case: language-code . . . . .	33
4.1.22	Test Case: strings . . . . .	35
4.1.23	Test Suite: structure . . . . .	36
4.1.24	Test Case: factory . . . . .	36
4.1.25	Test Suite: advanced-concept . . . . .	37
4.1.26	Test Case: t001 . . . . .	37
4.1.27	Test Case: t002 . . . . .	40
4.1.28	Test Case: t003 . . . . .	44
4.1.29	Test Case: t005 . . . . .	49
4.1.30	Test Case: t006 . . . . .	51
4.2	System Environment . . . . .	53
<b>5</b>	<b>Configuration Item Conclusion . . . . .</b>	<b>55</b>
	<b>Change Summary . . . . .</b>	<b>55</b>

## Approval Signatures

Role	Name	Signature
Author	Gary A. Howard	Electronic signature on file.
Project Lead	N/A	Electronic signature on file.
Technical Lead	N/A	Electronic signature on file.
System Engineer	N/A	Electronic signature on file.
Software Engineer	N/A	Electronic signature on file.
Test Engineer	N/A	Electronic signature on file.
Product Manager	N/A	Electronic signature on file.
Clinical App. Manager	N/A	Electronic signature on file.
RA Specialist	N/A	Electronic signature on file.
QA Engineer	Dayn A. Howard	Electronic signature on file.
Medical Doctor	N/A	Electronic signature on file.
Technical Reviewer	Cj Howard	Electronic signature on file.
Technical Reviewer	N/A	Electronic signature on file.
Technical Reviewer	N/A	Electronic signature on file.
Technical Reviewer	N/A	Electronic signature on file.
Independent Reviewer	N/A	Signature attached.

## 1 Introduction

### 1.1 Overview

This report demonstrates the Amber driver consumes YAML Test Plans, Test Suites, and Test Cases and produces output that is properly formatted for reports designed to tlc-article and audotoc conventions.

### 1.2 Purpose

This Report Package is a detailed record that provides a Configuration Item overview, a list of its Intended Use Requirements (IUR), one or more Test Reports, evidence the Test Reports ran, along with the output produced by the Test Report. The Test Report includes a pass/fail result for each Test Step and Test Report, a statement indicating the Configuration Item has a Configuration Identification and a conclusion that the Configuration Item has been validated for its intended use.

### 1.3 Scope

This Report Package applies to Company medical device software projects that have determined a Configuration Item must be validated for its intended use. This Report Package covers activities associated with validating a Configuration Item for its intended use requirements.

### 1.4 Deviations

The process governing the creation of this protocol and report deviates from the normal standard operating procedure (SOP005 Validation of Computerized Systems). This document combines both the protocol and the report. Normally the protocol is released first and report is released after the protocol is executed. This document represents an automated protocol execution facilitated through the use of automation scripting and software. The review of a paper protocol and pre-approval of said protocol does not satisfy the need to review the automated components used for the generation of this document. As a result, the automated components which codify the actual test protocol are reviewed by a technical approver as this document and the components are developed. This technical approver is an approver of this document and their approval indicates the automated components effectively test the article under test to meet the intended use as specified in the user requirements.

Additionally data obtained from the execution of the protocol is collected and presented in the grey boxes as objective evidence from the automated test application. Normally this would not be presented

together with the protocol, but given this is an automated process in a combined document; this is an effective means of retaining and presenting the objective evidence for review and approval.

Finally, presenting the protocol and the report together allows for a single step automation process that can be easily maintained and re-executed. Re-execution is often desired due to changes to the article under test or changes to user needs.

## 1.5 Tool Validation Objectives

Document 123-VNV-056022 Validation Determination report provides a Determination of Validation decision tree and Determination of Level of Risk and Validation Rigor decision tree to aid a Development Team when assessing the need for validation. 123-VNV-056022 was updated to indicate this tool required Validation and the Level of Risk needed. The steps below describe the steps used to validate a tool.

1. Describe the intended use of the tool.
2. Set the purpose and scope for the tool validation effort.
3. Enumerate intended use requirements.
4. Disclose compliance criteria.
5. Define Tool validation acceptance criteria.
6. Identify responsible persons and their roles.
7. Document required deliverables.
8. Define specific test steps and test steps to confirm that the Tool's intended use requirements have been met.
9. Collect test evidence.
10. Record Tool validation conclusion.

## 1.6 General Terms

**Configuration Control** The systematic process for managing changes to and established baseline.

**Configuration Identification** A unique identifier used to associate a collection of software artifacts.

**Configuration Items** Software source code, executables, build scripts, and other software development and software test artifacts relevant to creating and maintaining a software project.

**Configuration Status Accounting** The recording and reporting of the information needed to effectively manage the software and documentation components of a software project.

**Report Package** A detailed record that provides a Configuration Item overview, a list of its Intended Use Requirements (IUR), one or more Test Reports, evidence the Test Reports ran along with the output produced by Test Report including a pass/fail result for each Test Step and Test Report, and a statement indicating the Configuration Item has a Configuration Identification, and conclusion that the Configuration Item has been validated for its intended use.

**Test Plan** A test plan is a collection of one or more test suites a tester has determined to use to challenge requirements.

**Test Suite** A test suite is a collection of one or more test cases a tester has determined to use to challenge requirements.

**Test Case** A test case is a set of conditions under which a tester will determine whether the test is working as it was originally established for it to do.

**Test Step** A unique test identifier with predetermined expectation, confirmation criteria, and pass/fail result.

**Test Report** A test report consists of Detailed instructions for the set-up, execution, and evaluation of results for a given test. The test protocol may include one or more test cases for which the steps of the protocol will repeat with different input data. Test cases are chosen to ensure that corner cases in the code and data structures are covered. A test protocol may be a script that is automatically run by the computer.

## 1.7 General Acronyms

**FDA** Food and Drug Administration

**IUR** Intended Use Requirements

**LMS** Learning Management System

**SOP** Standard Operating Procedure

**SOUP** Software Of Unknown Provenance

## 1.8 References

- SOP004 Software Development Procedure
- SOP003 Software Configuration Management
- SOP001 Good Documentation Practices
- SOP005 Validation of Computerized Systems
- SOP002 Change Management

## 1.9 Training

Company's training records are stored in the Quality Management System. Additional training is not required because this is an automated test that is executed by the automated testing platform. SOP004 Software Development Procedure provides training required to create, maintain, and execute this testing protocol.

## 1.10 Tool Validation Test Approach

This Test Plan describes a series of Test Suites, Test Cases, and Test Steps. When executed, each Test Step determines if the Configuration Item satisfies one or more system requirements. When a Test Step indicates that the system requirements are satisfied, the Test Step's result is "pass". Otherwise, the Test Step's result is "fail". The computer records all "pass" and "fail" results in the Test Plan record. The Configuration Item is considered verified when all Test Steps are executed and the Test Plan record contains no "fail" results. Each Test Step that results in a deviation, observation, incident, or failure shall be represented in the final report.

## 1.11 Configuration Management

When a Configuration Item is changed, we will review the manufacturer's release notes or our design history file (DHF) to determine if regression testing or adjustments to this Report Package is necessary. We will verify the changes do not impact product operation, product quality, or quality decision made prior to performing the upgrade.

## 1.12 Test Plan Instructions

This Test Plan describes Test Suites, Test Cases, and Test Steps that demonstrate how the Configuration Item satisfies the IUR. Each Test Plan describes any setup criteria needed to conduct the test. Each Test Plan contains a list of IUR's and the steps that demonstrate how the Configuration Item satisfies the IUR. Each Test Step is marked passed or failed as it is completed. Each Test Plan is marked passed when all Test Steps pass or failed if a single Test Step fails. Failures are addressed per SOP002 Change Management. This serves as a record of the completed test.

Test Plans are automatically run by the computer, generating a report in PDF format. This Report Package is reviewed prior to execution per SOP004 Software Development Procedure. The Report Package is routed and archived in the Quality Management System. When it becomes necessary to annotate a computer generated document SOP001 Good Documentation Practices must be followed.

### 1.13 Test Plan Storage and Review

This Test Plan is part of a Company's automated validation framework. The framework consists of following parts:

**L<sup>A</sup>T<sub>E</sub>X** files are used to provide an Abstract, Introduction, Intended Use Requirements, Test Plan Overview, Test Equipment, Configuration Item Validation, Conclusion, and Change Summary. L<sup>A</sup>T<sub>E</sub>X files are converted assembled into PDF documents. PDF documents are routed using the Company's document management system for approval.

**Ruby** software is used to run the automated framework to collect test evidence.

**Git** is used as the storage repository for L<sup>A</sup>T<sub>E</sub>X & YAML files, a Git pull-request is used to review the L<sup>A</sup>T<sub>E</sub>X & YAML files prior to use.

**Evidence** Test Plan output includes one Test Suite, Test Plan, and Test Step, and Test Evidence.

**YAML** files define the Test Plan, Test Suite, and Test Steps that are processed to generate test evidence.

## 2 Requirements

Intended-use requirements are defined using the following story format:

*As a <type of user>, I want <some goal> so that <some reason>*

### AMBER-IUR-001

As the designer, I want to demonstrate Amber can process Test Plans, Test Suites, and Test Cases so that I can produce a testing report.

### AMBER-IUR-002

As the designer, I want to demonstrate Amber can invoke an another executable program so that I can collect evidence for my test reports.

### AMBER-IUR-003

As the designer, I want to demonstrate Amber can accept command line options so that I can control the type of testing Amber conducts.

### AMBER-IUR-004

As the designer, I want to demonstrate Amber can support nested Test Suites and Test Cases so that test objects can be logically organized.

### AMBER-IUR-005

As the designer, I want to demonstrate Amber can substitute keywords when processing YAML files so that the maintenance of Test Plans, Test Suites, and Test Cases is minimized.

### AMBER-IUR-006

As the designer, I want to demonstrate Amber can support embedded  $\text{\LaTeX}$  enumerate and itemized commands so that Test Plans, Test Suites and Test Cases have beautifully typeset lists.

### 3 Test Plan Overview

This section describes Test Plans, Test Suites, Test Cases, and Test Steps that demonstrate how a Configuration Item satisfies the IUR. Each Test Plan describes any setup criteria needed to conduct the Test Steps. Each Test Plan contains a list of IUR's and the Test Steps that demonstrate how the Configuration Item satisfies the IUR. Each Test Step is marked passed or failed as it is completed. Each Test Plan is marked passed when all Test Steps pass or failed if a single Test Step fails. This serves as a record of completed Test Plans and Test Steps.

Each Test Plan is described in its own section. The order the Test Plans are listed is the order they are run. Each Test Plan defines:

<b>name</b>	Each Plan, Suite, and Case has a unique name.
<b>purpose</b>	Each Plan, Suite, and Case has a purpose.
<b>Test Steps</b>	Each step has a confirmation and expectation along with the command needed to challenge the IUR.
<b>Objective Evidence</b>	A record the Test Plan was run along with any evidence collected while the Test Steps were run.
<b>Traceability</b>	Suites and Cases are traced to an IUR that is challenged. IUR can be traced to multiple Suites and Cases.

Each Test Plan, Test Suite, Test Case, and Test Step has been designed to be run by the computer. However, a person may choose to manually run the Test Step, save the test results, and generate this test report as specified in the appropriate design documentation. The example below runs these commands:

1. git help
2. cat .gitconfig

The output from both commands are written to the system console.

```
1 plan:
2   name: A Test Plan Name
3   purpose: purpose of the plan
4
5 suite:
6   name: A Test Suite Name
7   purpose: a suite purpose
8   requirement: IUR01 and IUR02
9
10  - case:
11    name: A Test Case name
12    purpose: A Test Case purpose
13    steps:
14      - confirm: Confirm git help is written to the console output.
15        expectation: Git help is displayed.
16        command: git
17        argument: help
18
19      - confirm: Confirm .git config is written to the console.
20        expectation: .gitconfig is written to the console output.
21        command: cat
22        argument: .gitconfig
```



## 4 Test Evidence

The Company's automation framework assembles the content in this section. The section has one or more Test Plans, Test Suites, Test Cases, and Test Evidence. The evidence provided is used to conclude the Tool has met the Intended Use Requirements.

### 4.1 Test Plan: master

**Purpose:** ~~LaTeX~~. ~~LaTeX~~ This Test Plan demonstrates the Ruby gem Amber functions correctly. This Test plan shows Amber has met the intended-use requirements defined by the designer. This Test Plan includes the Test Suites listed below.

- Test Suite cli/options shows all permutations of command line options function correctly.
- Test Suite utility/substitution demonstrates how Amber substitutes values found in Test Plans, Test Suites, and Test Cases to simplify test maintenance.
- Test Suite tif/structure reveals concepts unique to a Test Input Factory
- Test Suite advanced-concept demonstrates embedded ~~LaTeX~~ commands throughout.

Amber Test Cases have been designed to showoff Amber concepts. There are many Test Steps that may report failure. In these cases, an operation system application is simply not installed. For example, most Linux systems come with the man program installed. Amber will issue a command 'which man' and report PASS if man is installed and FAIL if man is not installed. In the failed test case, Amber prints the environment path that was searched. In short, Amber simply ran the command it was instructed to run and captured the results. This is sufficient for Amber's validation purposes.

#### 4.1.1 Test Suite: options

**Purpose:** This test suite demonstrates Amber consumes and uses command line arguments properly.

#### 4.1.2 Test Case: browser

**Purpose:** This test case is used to demonstrate Amber properly uses the `-browser` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-browser`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.browser_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Browser'`

**Execution start:** Jul 16, 2022 11:47:27.550834

**Execution end:** Jul 16, 2022 11:47:27.843818

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Browser/}
2
3 Amber CLO Browser
4   --browser=Chrome
5     has been used from the command line.
6   --browser Chrome
7     has been used from the command line.
8   -bChrome
9     has been used from the command line.
10  --browser=Firefox
11    has been used from the command line.
12  --browser Firefox
13    has been used from the command line.
14  -bFirefox
15    has been used from the command line.
16  --browser=IE
17    has been used from the command line.
18  --browser IE
19    has been used from the command line.
20  -bIE
21    has been used from the command line.
22  --browser=Opra
23    has been used from the command line.
24  --browser Opra
25    has been used from the command line.
26  -bOpra
27    has been used from the command line.
28
29 Finished in 0.00445 seconds (files took 0.19548 seconds to load)
30 12 examples, 0 failures
31
32 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 536 / 863
    LOC (62.11%) covered.
```

#### 4.1.3 Test Case: case

**Purpose:** This test case is used to demonstrate Amber properly uses the `-case` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-case`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.case_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Case'`

**Execution start:** Jul 16, 2022 11:47:27.844424

**Execution end:** Jul 16, 2022 11:47:28.136920

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Case/}
2
3 Amber CLO Case
4   -c has not been used.
5   -cbar has been used from the command line.
6   -c foobar has been used from the command line.
7
8 Amber CLO Case
9   --case has not been used.
10  --case=foo has been used from the command line.
11  --case baz has been used from the command line.
12
13 Finished in 0.00349 seconds (files took 0.19566 seconds to load)
14 6 examples, 0 failures
15
16 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 522 / 863
   LOC (60.49%) covered.
```

#### 4.1.4 Test Case: default

**Purpose:** This test case is used to demonstrate Amber properly constructs a default Amber::Options object.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly constructs a default Amber::Options object.

**Expectation:** Rspec output shows Amber::Options object is initialized correctly.

**Command:** rspec -format documentation -e 'Amber CLO Defaults'

**Execution start:** Jul 16, 2022 11:47:28.137578

**Execution end:** Jul 16, 2022 11:47:28.422609

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Defaults/}
2
3 All examples were filtered out
4
5 Finished in 0.00015 seconds (files took 0.19577 seconds to load)
6 0 examples, 0 failures
7
8 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 500 / 863
   LOC (57.94%) covered.
```

#### 4.1.5 Test Case: environment

**Purpose:** This test case demonstrates that Amber can record the operational environment in which it was used.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber understands when to record the operational environment.

**Expectation:** Rspec output shows Amber's understanding of the environment option.

**Command:** `rspec -format documentation -e 'Amber CLO Environment'`

**Execution start:** Jul 16, 2022 11:47:28.423184

**Execution end:** Jul 16, 2022 11:47:28.715289

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Environment/}
2
3 Amber CLO Environment
4   no -e
5     has not been used.
6   -e
7     has been used from the command line.
8   --environment
9     has been used from the command line.
10
11 Amber CLO Environment
12   dollar_signs are escaped.
13
14 Finished in 0.00371 seconds (files took 0.19521 seconds to load)
15 4 examples, 0 failures
16
17 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 515 / 863
   LOC (59.68%) covered.
```

#### 4.1.6 Test Case: file

**Purpose:** This test case is used to demonstrate Amber properly uses the `-file` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-file`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.file_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO File'`

**Execution start:** Jul 16, 2022 11:47:28.715945

**Execution end:** Jul 16, 2022 11:47:29.011264

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ File/}
2
3 Amber CLO File Short
4   no -f has not been used.
5   -fa.yaml has been used from the command line.
6
7 Amber CLO File Long
8   --file has not been used.
9   --file=b.yaml has been used from the command line.
10  --file c.yaml has been used from the command line.
11
12 Finished in 0.00341 seconds (files took 0.19847 seconds to load)
13 5 examples, 0 failures
14
15 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 518 / 863
    LOC (60.02%) covered.
```

#### 4.1.7 Test Case: language

**Purpose:** This test case is used to demonstrate Amber properly uses the `-language` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-language`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.language_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Language'`

**Execution start:** Jul 16, 2022 11:47:29.011853

**Execution end:** Jul 16, 2022 11:47:29.317094

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Language/}
2
3 Amber CLO Language
4   no -l
5     has not been used.
6   with unknown language
7     --language XX
8       raises an invalid argument exception
9     --language=XX
10      raises an invalid argument exception
11   -l XX
12     raises an invalid argument exception
13   -lXX
14     raises an invalid argument exception
15 behaves like Amber CLO language parameter
16   --language=zz
17     returns n/a when run with double dash and equal sign
18   --language zz
19     returns n/a when run with double dash and a space
20   -lzz
21     returns n/a when run with dash and no space
22   -l zz
23     returns n/a when run with dash and a space
24 behaves like Amber CLO language parameter
25   --language=cs
26     returns Czech when run with double dash and equal sign
27   --language cs
28     returns Czech when run with double dash and a space
29   -lcs
30     returns Czech when run with dash and no space
31   -l cs
32     returns Czech when run with dash and a space
33 behaves like Amber CLO language parameter
34   --language=cy
35     returns Welsh when run with double dash and equal sign
36   --language cy
37     returns Welsh when run with double dash and a space
38   -lcy
39     returns Welsh when run with dash and no space
40   -l cy
41     returns Welsh when run with dash and a space
42 behaves like Amber CLO language parameter
43   --language=da
44     returns Danish when run with double dash and equal sign
45   --language da
46     returns Danish when run with double dash and a space
47   -lda
```

```
48     returns Danish when run with dash and no space
49     -l da
50     returns Danish when run with dash and a space
51 behaves like Amber CLO language parameter
52     --language=de
53     returns German when run with double dash and equal sign
54     --language de
55     returns German when run with double dash and a space
56     -lde
57     returns German when run with dash and no space
58     -l de
59     returns German when run with dash and a space
60 behaves like Amber CLO language parameter
61     --language=en
62     returns English when run with double dash and equal sign
63     --language en
64     returns English when run with double dash and a space
65     -len
66     returns English when run with dash and no space
67     -l en
68     returns English when run with dash and a space
69 behaves like Amber CLO language parameter
70     --language=es
71     returns Spanish; Castilian when run with double dash and equal sign
72     --language es
73     returns Spanish; Castilian when run with double dash and a space
74     -les
75     returns Spanish; Castilian when run with dash and no space
76     -l es
77     returns Spanish; Castilian when run with dash and a space
78 behaves like Amber CLO language parameter
79     --language=fi
80     returns Finnish when run with double dash and equal sign
81     --language fi
82     returns Finnish when run with double dash and a space
83     -lfi
84     returns Finnish when run with dash and no space
85     -l fi
86     returns Finnish when run with dash and a space
87 behaves like Amber CLO language parameter
88     --language=fr
89     returns French when run with double dash and equal sign
90     --language fr
91     returns French when run with double dash and a space
92     -lfr
93     returns French when run with dash and no space
94     -l fr
95     returns French when run with dash and a space
96 behaves like Amber CLO language parameter
97     --language=fr-ca
98     returns CA French - Canadian when run with double dash and equal sign
99     --language fr-ca
100    returns CA French - Canadian when run with double dash and a space
101    -lfr-ca
102    returns CA French - Canadian when run with dash and no space
103    -l fr-ca
104    returns CA French - Canadian when run with dash and a space
105 behaves like Amber CLO language parameter
106    --language=fr-eu
107    returns EU French - European when run with double dash and equal sign
108    --language fr-eu
109    returns EU French - European when run with double dash and a space
110    -lfr-eu
111    returns EU French - European when run with dash and no space
112    -l fr-eu
113    returns EU French - European when run with dash and a space
114 behaves like Amber CLO language parameter
115    --language=fy
116    returns Western Frisian when run with double dash and equal sign
117    --language fy
118    returns Western Frisian when run with double dash and a space
119    -lfy
120    returns Western Frisian when run with dash and no space
```



```
121 -l fy
122     returns Western Frisian when run with dash and a space
123 behaves like Amber CLO language parameter
124 --language=it
125     returns Italian when run with double dash and equal sign
126 --language it
127     returns Italian when run with double dash and a space
128 -lit
129     returns Italian when run with dash and no space
130 -l it
131     returns Italian when run with dash and a space
132 behaves like Amber CLO language parameter
133 --language=nl
134     returns Dutch; Flemish when run with double dash and equal sign
135 --language nl
136     returns Dutch; Flemish when run with double dash and a space
137 -lnl
138     returns Dutch; Flemish when run with dash and no space
139 -l nl
140     returns Dutch; Flemish when run with dash and a space
141 behaves like Amber CLO language parameter
142 --language=no
143     returns Norwegian when run with double dash and equal sign
144 --language no
145     returns Norwegian when run with double dash and a space
146 -lno
147     returns Norwegian when run with dash and no space
148 -l no
149     returns Norwegian when run with dash and a space
150 behaves like Amber CLO language parameter
151 --language=pl
152     returns Polish when run with double dash and equal sign
153 --language pl
154     returns Polish when run with double dash and a space
155 -lpl
156     returns Polish when run with dash and no space
157 -l pl
158     returns Polish when run with dash and a space
159 behaves like Amber CLO language parameter
160 --language=pt
161     returns Portuguese when run with double dash and equal sign
162 --language pt
163     returns Portuguese when run with double dash and a space
164 -lpt
165     returns Portuguese when run with dash and no space
166 -l pt
167     returns Portuguese when run with dash and a space
168 behaves like Amber CLO language parameter
169 --language=ro
170     returns Romanian; Moldavian; Moldovan when run with double dash and equal sign
171 --language ro
172     returns Romanian; Moldavian; Moldovan when run with double dash and a space
173 -lro
174     returns Romanian; Moldavian; Moldovan when run with dash and no space
175 -l ro
176     returns Romanian; Moldavian; Moldovan when run with dash and a space
177 behaves like Amber CLO language parameter
178 --language=ru
179     returns Russian when run with double dash and equal sign
180 --language ru
181     returns Russian when run with double dash and a space
182 -lru
183     returns Russian when run with dash and no space
184 -l ru
185     returns Russian when run with dash and a space
186 behaves like Amber CLO language parameter
187 --language=sk
188     returns Slovak when run with double dash and equal sign
189 --language sk
190     returns Slovak when run with double dash and a space
191 -lsk
192     returns Slovak when run with dash and no space
193 -l sk
```

```
194     returns Slovak when run with dash and a space
195 behaves like Amber CLO language parameter
196   --language=sv
197     returns Swedish when run with double dash and equal sign
198   --language sv
199     returns Swedish when run with double dash and a space
200   -lsv
201     returns Swedish when run with dash and no space
202   -l sv
203     returns Swedish when run with dash and a space
204 behaves like Amber CLO language parameter
205   --language=zu
206     returns Zulu when run with double dash and equal sign
207   --language zu
208     returns Zulu when run with double dash and a space
209   -lzu
210     returns Zulu when run with dash and no space
211   -l zu
212     returns Zulu when run with dash and a space
213
214 Finished in 0.01686 seconds (files took 0.19489 seconds to load)
215 93 examples, 0 failures
216
217 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 531 / 863
    LOC (61.53%) covered.
```

#### 4.1.8 Test Case: nodryrun

**Purpose:** This test case is used to demonstrate Amber properly uses the `--nodryrun` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `--nodryrun`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.nodryrun_option` handles supported argument formats.

**Command:** `rspec --format documentation -e 'Amber CLO NoDryRun'`

**Execution start:** Jul 16, 2022 11:47:29.317771

**Execution end:** Jul 16, 2022 11:47:29.613725

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ NoDryRun/}
2
3 Amber CLO NoDryRun
4   no --n
5     has not been used.
6   --n
7     has been used from the command line.
8   --nodryrun
9     has been used from the command line.
10
11 Finished in 0.00343 seconds (files took 0.19922 seconds to load)
12 3 examples, 0 failures
13
14 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 508 / 863
    LOC (58.86%) covered.
```

#### 4.1.9 Test Case: obliterate

**Purpose:** This test case is used to demonstrate Amber properly uses the `-obliterate` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-obliterate`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.obliterate_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Obliterate'`

**Execution start:** Jul 16, 2022 11:47:29.614336

**Execution end:** Jul 16, 2022 11:47:29.906590

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Obliterate/}
2
3 Amber CLO Obliterate
4   no -O
5     has not been used.
6   -O
7     has been used from the command line.
8   --obliterate
9     has been used from the command line.
10
11 Finished in 0.00333 seconds (files took 0.19477 seconds to load)
12 3 examples, 0 failures
13
14 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 508 / 863
    LOC (58.86%) covered.
```

#### 4.1.10 Test Case: plan

**Purpose:** This test case is used to demonstrate Amber properly uses the `-plan` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-plan`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.plan_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Plan'`

**Execution start:** Jul 16, 2022 11:47:29.907264

**Execution end:** Jul 16, 2022 11:47:30.199331

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Plan/}
2
3 Amber CLO Plan Short
4   -p has not been used.
5   -pbar has been used from the command line.
6   -p foobar has been used from the command line.
7
8 Amber CLO Plan Long
9   --plan has not been used.
10  --plan=foo has been used from the command line.
11  --plan baz has been used from the command line.
12
13 Finished in 0.00375 seconds (files took 0.1953 seconds to load)
14 6 examples, 0 failures
15
16 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 522 / 863
   LOC (60.49%) covered.
```

#### 4.1.11 Test Case: simulate

**Purpose:** This test case is used to demonstrate Amber properly uses the `-simulate` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-simulate`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.simulate_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Simulate'`

**Execution start:** Jul 16, 2022 11:47:30.200051

**Execution end:** Jul 16, 2022 11:47:30.490941

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Simulate/}
2
3 Amber CLO Simulate
4   no -S
5     has not been used.
6   -S
7     has been used from the command line.
8   --simulate
9     has been used from the command line.
10
11 Finished in 0.00336 seconds (files took 0.194 seconds to load)
12 3 examples, 0 failures
13
14 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 508 / 863
    LOC (58.86%) covered.
```

#### 4.1.12 Test Case: suite

**Purpose:** This test case is used to demonstrate Amber properly uses the `--suite` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `--suite`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.suite_option` handles supported argument formats.

**Command:** `rspec --format documentation -e 'Amber CLO Suite'`

**Execution start:** Jul 16, 2022 11:47:30.491526

**Execution end:** Jul 16, 2022 11:47:30.783326

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Suite/}
2
3 Amber CLO Suite Short
4   -s has not been used.
5   -sbar has been used from the command line.
6   -s foobar has been used from the command line.
7
8 Amber CLO Suite Long
9   --suite has not been used.
10  --suite=foo has been used from the command line.
11  --suite baz has been used from the command line.
12
13 Finished in 0.00368 seconds (files took 0.19471 seconds to load)
14 6 examples, 0 failures
15
16 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 522 / 863
   LOC (60.49%) covered.
```

#### 4.1.13 Test Case: verbose

**Purpose:** This test case is used to demonstrate Amber properly uses the `-verbose` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-verbose`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.verbose_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Verbose'`

**Execution start:** Jul 16, 2022 11:47:30.783993

**Execution end:** Jul 16, 2022 11:47:31.074470

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Verbose/}
2
3 Amber CLO Verbose
4   no -v
5     has not been used.
6   -v
7     has been used from the command line.
8   --verbose
9     has been used from the command line.
10
11 Finished in 0.00334 seconds (files took 0.19387 seconds to load)
12 3 examples, 0 failures
13
14 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 508 / 863
    LOC (58.86%) covered.
```



#### 4.1.14 Test Case: version

**Purpose:** This test case is used to demonstrate Amber properly uses the `–version` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `–version`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.version_option` handles supported argument formats.

**Command:** `rspec –format documentation -e 'Amber CLO Version'`

**Execution start:** Jul 16, 2022 11:47:31.075059

**Execution end:** Jul 16, 2022 11:47:31.366651

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Version/}
2
3 Amber CLO Version
4   no —version
5     was not used. However the version number must match 1.6.4.412
6   —version
7 1.6.4.412
8     has been used from the command line.
9   Version
10     has a version number
11     version number must match 1.6.4.412
12
13 Finished in 0.00374 seconds (files took 0.1951 seconds to load)
14 4 examples, 0 failures
15
16 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 507 / 863
    LOC (58.75%) covered.
```

#### 4.1.15 Test Case: writer

**Purpose:** This test case is used to demonstrate Amber properly uses the `-writer` command line option.

**Requirement:** AMBER-IUR-001, AMBER-IUR-002 and AMBER-IUR-003

**Step:** 1

**Confirm:** Amber properly consumes the command line argument `-writer`.

**Expectation:** Rspec output shows `Amber::CommandLineOptions.writer_option` handles supported argument formats.

**Command:** `rspec -format documentation -e 'Amber CLO Writer'`

**Execution start:** Jul 16, 2022 11:47:31.367302

**Execution end:** Jul 16, 2022 11:47:31.658313

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Amber\ CLO\ Writer/}
2
3 Amber CLO Writer Short
4   -w has not been used.
5   -wAscii has been used from the command line.
6   -w Ascii has been used from the command line.
7   -wLaTeX has been used from the command line.
8   -w LaTeX has been used from the command line.
9
10 Amber CLO Writer Long
11   --writer has not been used.
12   --writer=Ascii has been used from the command line.
13   --writer Ascii has been used from the command line.
14   --writer=LaTeX has been used from the command line.
15   --writer LaTeX has been used from the command line.
16
17 Finished in 0.00381 seconds (files took 0.19461 seconds to load)
18 10 examples, 0 failures
19
20 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 528 / 863
   LOC (61.18%) covered.
```

#### 4.1.16 Test Suite: substitute

**Purpose:** This test suite demonstrates Amber's runtime substitution capabilities. Amber has been designed to translate the keywords below.

1. browser or BROWSER
2. file or FILE
3. language or LANGUAGE
4. language-code or LANGUAGE-CODE
5. home or HOME or ~

This Test case also demonstrated embedded L<sup>A</sup>T<sub>E</sub>X syntax to define the list above.

#### 4.1.17 Test Case: browser

**Purpose:** This test case is used to demonstrate the browser keyword is properly substituted by Amber.

**Requirement:** AMBER-IUR-004 and AMBER-IUR-005

**Step:** 1

**Confirm:** Amber properly substitutes the browser keyword for all browser types.

**Expectation:** RSpec output shows Amber::Substitute.browser properly substituted browser and BROWSER keywords to Chrome, Firefox, Edge, and IE.

**Command:** rspec -format documentation -e 'YAML Browser Substitutions'

**Execution start:** Jul 16, 2022 11:47:31.949264

**Execution end:** Jul 16, 2022 11:47:32.245261

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/YAML\ Browser\ Substitutions/}
2
3 YAML Browser Substitutions
4   Amber::Substitute.browser
5     can substitute ${BROWSER} to None
6     can substitute ${browser} to None
7
8 YAML Browser Substitutions
9   Amber::Substitute.browser
10    can substitute ${BROWSER} to Brave
11
12 YAML Browser Substitutions
13   Amber::Substitute.browser
14    can substitute ${BROWSER} to Chrome
15
16 YAML Browser Substitutions
17   Amber::Substitute.browser
18    can substitute ${browser} to Edge
19
20 YAML Browser Substitutions
21   Amber::Substitute.browser
22    can substitute ${BROWSER} to Firefox
23
24 YAML Browser Substitutions
25   Amber::Substitute.browser
26    can substitute ${browser} to IE
27
28 YAML Browser Substitutions
29   Amber::Substitute.browser
30    can substitute ${BROWSER} to Opera
31
```

```
32 Finished in 0.0043 seconds (files took 0.1978 seconds to load)
33 8 examples, 0 failures
34
35 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 522 / 863
    LOC (60.49%) covered.
```

#### 4.1.18 Test Case: extend-path

**Purpose:** This test case is used to demonstrate the tilde marker is properly substituted by Amber.

**Requirement:** AMBER-IUR-004 and AMBER-IUR-005

**Step:** 1

**Confirm:** Amber properly substitutes the tilde marker correctly for the operating system.

**Expectation:** Rspec output shows Amber::Substitute.extend\_path substituted ~ to the home directory.

**Command:** rspec -format documentation -e 'YAML Extend Path Substitutions'

**Execution start:** Jul 16, 2022 11:47:32.245914

**Execution end:** Jul 16, 2022 11:47:32.539978

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/YAML\ Extend\ Path\ Substitutions/}
2
3 YAML Extend Path Substitutions
4   Amber::Substitute.expected_path
5     can expand ~ to /home/traap
6     can expand ~ and ~ to /home/traap and /home/traap
7
8 Finished in 0.00301 seconds (files took 0.19589 seconds to load)
9 2 examples, 0 failures
10
11 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 502 / 863
    LOC (58.17%) covered.
```

#### 4.1.19 Test Case: home

**Purpose:** This test case is used to demonstrate the home keyword is properly substituted by Amber.

**Requirement:** AMBER-IUR-004 and AMBER-IUR-005

**Step:** 1

**Confirm:** Amber properly substitutes the home keyword for all browser types.

**Expectation:** Rpec output shows Amber::Substitute.home properly substituted home and HOME keywords specific to this operating system.

**Command:** rspec -format documentation -e 'YAML Home Substitutions'

**Execution start:** Jul 16, 2022 11:47:32.540650

**Execution end:** Jul 16, 2022 11:47:32.833851

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/YAML\ Home\ Substitutions/}
2
3 YAML Home Substitutions
4   Amber::Substitute.home
5     can substitute ${home} to ~
6     can substitute ${HOME} to ~
7     can substitute ${home} and ${HOME} to ~ and ~
8
9 Finished in 0.00336 seconds (files took 0.19633 seconds to load)
10 3 examples, 0 failures
11
12 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 504 / 863
    LOC (58.4%) covered.
```

#### 4.1.20 Test Case: language

**Purpose:** This test case is used to demonstrate the language keyword is properly substituted by Amber.

**Requirement:** AMBER-IUR-004 and AMBER-IUR-005

**Step:** 1

**Confirm:** Amber properly substitutes the language keyword for all supported languages.

**Expectation:** Rspec output shows Amber::Substitute.language properly substituted language and LANGUAGE keywords to zz, cs, da, de, en, es, fr-ca, fr-eu, it, ne, no, pl, so, and sv.

**Command:** rspec -format documentation -e 'YAML Language Substitutions'

**Execution start:** Jul 16, 2022 11:47:32.834469

**Execution end:** Jul 16, 2022 11:47:33.131859

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/YAML\ Language\ Substitutions/}
2
3 YAML Language Substitutions
4   behaves like Amber::Substitute.language
5     can substitute ${language} to n/a
6     can substitute ${LANGUAGE} to n/a
7   behaves like Amber::Substitute.language
8     can substitute ${language} to Czech
9     can substitute ${LANGUAGE} to Czech
10  behaves like Amber::Substitute.language
11    can substitute ${language} to Welsh
12    can substitute ${LANGUAGE} to Welsh
13  behaves like Amber::Substitute.language
14    can substitute ${language} to Danish
15    can substitute ${LANGUAGE} to Danish
16  behaves like Amber::Substitute.language
17    can substitute ${language} to German
18    can substitute ${LANGUAGE} to German
19  behaves like Amber::Substitute.language
20    can substitute ${language} to English
21    can substitute ${LANGUAGE} to English
22  behaves like Amber::Substitute.language
23    can substitute ${language} to Spanish; Castilian
24    can substitute ${LANGUAGE} to Spanish; Castilian
25  behaves like Amber::Substitute.language
26    can substitute ${language} to Finnish
27    can substitute ${LANGUAGE} to Finnish
28  behaves like Amber::Substitute.language
29    can substitute ${language} to French
30    can substitute ${LANGUAGE} to French
31  behaves like Amber::Substitute.language
32    can substitute ${language} to CA French - Canadian
33    can substitute ${LANGUAGE} to CA French - Canadian
34  behaves like Amber::Substitute.language
35    can substitute ${language} to EU French - European
36    can substitute ${LANGUAGE} to EU French - European
37  behaves like Amber::Substitute.language
38    can substitute ${language} to Western Frisian
39    can substitute ${LANGUAGE} to Western Frisian
40  behaves like Amber::Substitute.language
41    can substitute ${language} to Italian
42    can substitute ${LANGUAGE} to Italian
43  behaves like Amber::Substitute.language
44    can substitute ${language} to Dutch; Flemish
45    can substitute ${LANGUAGE} to Dutch; Flemish
46  behaves like Amber::Substitute.language
```

```
47     can substitute ${language} to Norwegian
48     can substitute ${LANGUAGE} to Norwegian
49 behaves like Amber::Substitute.language
50     can substitute ${language} to Polish
51     can substitute ${LANGUAGE} to Polish
52 behaves like Amber::Substitute.language
53     can substitute ${language} to Portuguese
54     can substitute ${LANGUAGE} to Portuguese
55 behaves like Amber::Substitute.language
56     can substitute ${language} to Romanian; Moldavian; Moldovan
57     can substitute ${LANGUAGE} to Romanian; Moldavian; Moldovan
58 behaves like Amber::Substitute.language
59     can substitute ${language} to Russian
60     can substitute ${LANGUAGE} to Russian
61 behaves like Amber::Substitute.language
62     can substitute ${language} to Slovak
63     can substitute ${LANGUAGE} to Slovak
64 behaves like Amber::Substitute.language
65     can substitute ${language} to Swedish
66     can substitute ${LANGUAGE} to Swedish
67 behaves like Amber::Substitute.language
68     can substitute ${language} to Zulu
69     can substitute ${LANGUAGE} to Zulu
70
71 Finished in 0.00706 seconds (files took 0.19676 seconds to load)
72 44 examples, 0 failures
73
74 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 502 / 863
    LOC (58.17%) covered.
```



#### 4.1.21 Test Case: language-code

**Purpose:** This test case is used to demonstrate the language-code keyword is properly substituted by Amber.

**Requirement:** AMBER-IUR-004 and AMBER-IUR-005

**Step:** 1

**Confirm:** Amber properly substitutes the language-code keyword for all supported languages.

**Expectation:** Rspec output shows Amber::Substitute.language-code properly substituted language-code and LANGUAGE-CODE keywords to n/a, Czech, Dansk, Deutsch, English, Espanol, CA French - Canadian, EU French - European, Italiano, Nederlands, Norsk, Polish, Romanian, and Svenska.

**Command:** rspec -format documentation -e 'YAML Language Code Substitutions'

**Execution start:** Jul 16, 2022 11:47:33.132591

**Execution end:** Jul 16, 2022 11:47:33.430465

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/YAML\ Language\ Code\ Substitutions/}
2
3 YAML Language Code Substitutions
4   behaves like Amber::Substitute.language_code
5     substitutes ${LANGUAGE-CODE} to zz
6     substitutes ${langauge-code} to zz
7   behaves like Amber::Substitute.language_code
8     substitutes ${LANGUAGE-CODE} to cs
9     substitutes ${langauge-code} to cs
10  behaves like Amber::Substitute.language_code
11    substitutes ${LANGUAGE-CODE} to cy
12    substitutes ${langauge-code} to cy
13  behaves like Amber::Substitute.language_code
14    substitutes ${LANGUAGE-CODE} to da
15    substitutes ${langauge-code} to da
16  behaves like Amber::Substitute.language_code
17    substitutes ${LANGUAGE-CODE} to de
18    substitutes ${langauge-code} to de
19  behaves like Amber::Substitute.language_code
20    substitutes ${LANGUAGE-CODE} to en
21    substitutes ${langauge-code} to en
22  behaves like Amber::Substitute.language_code
23    substitutes ${LANGUAGE-CODE} to es
24    substitutes ${langauge-code} to es
25  behaves like Amber::Substitute.language_code
26    substitutes ${LANGUAGE-CODE} to fi
27    substitutes ${langauge-code} to fi
28  behaves like Amber::Substitute.language_code
29    substitutes ${LANGUAGE-CODE} to fr
30    substitutes ${langauge-code} to fr
31  behaves like Amber::Substitute.language_code
32    substitutes ${LANGUAGE-CODE} to fr-ca
33    substitutes ${langauge-code} to fr-ca
34  behaves like Amber::Substitute.language_code
35    substitutes ${LANGUAGE-CODE} to fr-eu
36    substitutes ${langauge-code} to fr-eu
37  behaves like Amber::Substitute.language_code
38    substitutes ${LANGUAGE-CODE} to fy
39    substitutes ${langauge-code} to fy
40  behaves like Amber::Substitute.language_code
41    substitutes ${LANGUAGE-CODE} to it
42    substitutes ${langauge-code} to it
43  behaves like Amber::Substitute.language_code
44    substitutes ${LANGUAGE-CODE} to nl
45    substitutes ${langauge-code} to nl
```

```
46 behaves like Amber::Substitute.language_code
47   substitutes ${LANGUAGE-CODE} to no
48   substitutes ${langauge-code} to no
49 behaves like Amber::Substitute.language_code
50   substitutes ${LANGUAGE-CODE} to pl
51   substitutes ${langauge-code} to pl
52 behaves like Amber::Substitute.language_code
53   substitutes ${LANGUAGE-CODE} to pt
54   substitutes ${langauge-code} to pt
55 behaves like Amber::Substitute.language_code
56   substitutes ${LANGUAGE-CODE} to ro
57   substitutes ${langauge-code} to ro
58 behaves like Amber::Substitute.language_code
59   substitutes ${LANGUAGE-CODE} to ru
60   substitutes ${langauge-code} to ru
61 behaves like Amber::Substitute.language_code
62   substitutes ${LANGUAGE-CODE} to sk
63   substitutes ${langauge-code} to sk
64 behaves like Amber::Substitute.language_code
65   substitutes ${LANGUAGE-CODE} to sv
66   substitutes ${langauge-code} to sv
67 behaves like Amber::Substitute.language_code
68   substitutes ${LANGUAGE-CODE} to zu
69   substitutes ${langauge-code} to zu
70
71 Finished in 0.00697 seconds (files took 0.19701 seconds to load)
72 44 examples, 0 failures
73
74 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 502 / 863
   LOC (58.17%) covered.
```

#### 4.1.22 Test Case: strings

**Purpose:** This test case is used to demonstrate the Amber substitutes multiple keywords in data stream.

**Requirement:** AMBER-IUR-004 and AMBER-IUR-005

**Step:** 1

**Confirm:** Amber properly substitutes all keywords in a data stream.

**Expectation:** Rspec output shows Amber::Substitute.strings substituted all keywords without encountering an error.

**Command:** rspec -format documentation -e 'YAML Strings Substitutions'

**Execution start:** Jul 16, 2022 11:47:33.431068

**Execution end:** Jul 16, 2022 11:47:33.724930

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/YAML\ Strings\ Substitutions/}
2
3 YAML Strings Substitutions
4   Amber::Substitute.strings
5     can substitute ${BROWSER} to Opera
6     can substitute ${browser} ${file} ${language} and ${language-code} to Opera baz
7     can substitute ${language-code}${file}${language}${browser} to svbazSwedishOpera
8
9 Finished in 0.00313 seconds (files took 0.19681 seconds to load)
10 3 examples, 0 failures
11
12 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 503 / 863
    LOC (58.29%) covered.
```

#### 4.1.23 Test Suite: structure

**Purpose:** This test suite demonstrated Amber's ability to locate a nested Test Plan, Test Suite, or Test Case YAML file.

#### 4.1.24 Test Case: factory

**Purpose:** This test case is used to demonstrate Amber can properly locate a nested Test Plan, Test Suite, or Test Case.

**Requirement:** AMBER-IUR-004

**Step:** 1

**Confirm:** Amber properly locates nested Test Plan, Test Suite, and Test Case names.

**Expectation:** Rspec output shows Amber::FactoryStructure properly locates the YAML file below.

1. Test Plan foo
2. Nested Test Plan baz
3. Test Suite foo
4. Nested Test Suite name baz
5. Test Case foo
6. Nested Test Case name baz

**Command:** `rspec -format documentation -e 'Factory Structure'`

**Execution start:** Jul 16, 2022 11:47:34.012573

**Execution end:** Jul 16, 2022 11:47:34.310517

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 Run options: include {:full_description=>/Factory\ Structure/}
2
3 Factory Structure
4   Test Plan Name
5     is normal.
6     is nested.
7
8 Factory Structure
9   Test Suite Name
10    is normal.
11    is nested.
12
13 Factory Structure
14   Test Case Name
15     is normal.
16     is nested.
17
18 Finished in 0.00389 seconds (files took 0.19868 seconds to load)
19 6 examples, 0 failures
20
21 Coverage report generated for Unit Tests to /home/traap/git/amber/coverage. 512 / 863
    LOC (59.33%) covered.
```

#### 4.1.25 Test Suite: advanced-concept

**Purpose:** This Test Suite demonstrates a future concept that might be implemented. The concepts include the items below.

1. setup-before-all are Test Steps that are run before all Test Cases.
2. setup-before-each are Test Steps that are run before each Test Case.
3. teardown-after-all are Test Steps that are run before all Test Cases.
4. teardown-after-each are Test Steps that are run before each Test Case.

This Test Suite does demonstrate embedded  $\LaTeX$  commands and it includes the following Test Cases.

- t001
- t002
- t003
- t005
- t006

**Requirement:** AMBER-IUR-006

#### 4.1.26 Test Case: t001

**Purpose:** This Test Case demonstrates embedded  $\LaTeX$  enumerate and itemized commands.

1. Step #1 uses the Linux echo command.
2. Step #2 will use Linux date command.
3. Step #3 confirms the Linux curl program is installed.

**Requirement:** AMBER-IUR-006

**Step:** 1

**Confirm:** • Program echo has been installed.

**Expectation:** • echo installation location is displayed.

**Command:** sudo which echo

**Execution start:** Jul 16, 2022 11:47:34.597607

**Execution end:** Jul 16, 2022 11:47:37.954709

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/echo
```

**Step: 2**

**Confirm:**   • Program date has been installed.

**Expectation:**   • date installation location is displayed.

**Command:** which date

**Execution start:** Jul 16, 2022 11:47:37.955977

**Execution end:** Jul 16, 2022 11:47:37.958161

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/date
```

**Step: 3**

**Confirm:**   • Program curl has been installed.

**Expectation:**   • curl installation location is displayed.

**Command:** which curl

**Execution start:** Jul 16, 2022 11:47:37.959265

**Execution end:** Jul 16, 2022 11:47:37.961377

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/curl
```

#### 4.1.27 Test Case: t002

**Purpose:** This Test Case demonstrates embedded  $\LaTeX$  enumerate command.

1. Step #1 grep check
2. Step #2 openssl check
3. Step #3 sed check
4. Step #4 tr check

**Requirement:** AMBER-IUR-006

**Step:** 1

**Confirm:** Program grep has been installed.

**Expectation:** grep installation location is displayed.

**Command:** sudo which grep

**Execution start:** Jul 16, 2022 11:47:37.964705

**Execution end:** Jul 16, 2022 11:47:38.023167

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/grep
```



**Step: 2**

**Confirm:** Program openssl has been installed.

**Expectation:** openssl installation location is displayed.

**Command:** which openssl

**Execution start:** Jul 16, 2022 11:47:38.024013

**Execution end:** Jul 16, 2022 11:47:38.025469

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/openssl
```

**Step: 3**

**Confirm:** Program sed has been installed.

**Expectation:** sed installation location is displayed.

**Command:** which sed

**Execution start:** Jul 16, 2022 11:47:38.026204

**Execution end:** Jul 16, 2022 11:47:38.027650

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/sed
```

**Step: 4**

**Confirm:** Program tr has been installed.

**Expectation:** tr installation location is displayed.

**Command:** which tr

**Execution start:** Jul 16, 2022 11:47:38.028381

**Execution end:** Jul 16, 2022 11:47:38.029921

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/tr
```

#### 4.1.28 Test Case: t003

**Purpose:** This Test Case demonstrates embedded  $\LaTeX$  enumerate command.

1. Step #1 nice check
2. Step #2 nl check
3. Step #3 bzmores check
4. Step #4 latexmk check
5. Step #5 git check

**Requirement:** AMBER-IUR-006

**Step:** 1

**Confirm:** Program nice has installed.

**Expectation:** nice installation location is displayed.

**Command:** sudo which nice

**Execution start:** Jul 16, 2022 11:47:38.032342

**Execution end:** Jul 16, 2022 11:47:38.079884

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/nice
```

**Step: 2**

**Confirm:** Program nl has installed.

**Expectation:** nl installation location is displayed.

**Command:** which nl

**Execution start:** Jul 16, 2022 11:47:38.080850

**Execution end:** Jul 16, 2022 11:47:38.082342

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/nl
```

**Step: 3**

**Confirm:** Program which bzmre has been installed.

**Expectation:** bzmre installation location is displayed.

**Command:** which bzmre

**Execution start:** Jul 16, 2022 11:47:38.083281

**Execution end:** Jul 16, 2022 11:47:38.084860

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/bzmre
```

**Step: 4**

**Confirm:** Program latexmk has been installed.

**Expectation:** latexmk installation location is displayed.

**Command:** which latexmk

**Execution start:** Jul 16, 2022 11:47:38.085601

**Execution end:** Jul 16, 2022 11:47:38.087032

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/latexmk
```

**Step: 5**

**Confirm:** A developer is able to access Git help.

**Expectation:** Git help is displayed.

**Command:** git help

**Execution start:** Jul 16, 2022 11:47:38.087820

**Execution end:** Jul 16, 2022 11:47:38.090206

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
2           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
3           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
4           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
5           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
6           <command> [<args>]
7
8 These are common Git commands used in various situations:
9
10 start a working area (see also: git help tutorial)
11   clone      Clone a repository into a new directory
12   init       Create an empty Git repository or reinitialize an existing one
13
14 work on the current change (see also: git help everyday)
15   add        Add file contents to the index
16   mv         Move or rename a file, a directory, or a symlink
17   restore    Restore working tree files
18   rm         Remove files from the working tree and from the index
19
20 examine the history and state (see also: git help revisions)
21   bisect     Use binary search to find the commit that introduced a bug
22   diff       Show changes between commits, commit and working tree, etc
23   grep       Print lines matching a pattern
24   log        Show commit logs
25   show       Show various types of objects
26   status     Show the working tree status
27
28 grow, mark and tweak your common history
29   branch     List, create, or delete branches
30   commit     Record changes to the repository
31   merge      Join two or more development histories together
32   rebase     Reapply commits on top of another base tip
33   reset      Reset current HEAD to the specified state
34   switch     Switch branches
35   tag        Create, list, delete or verify a tag object signed with GPG
36
37 collaborate (see also: git help workflows)
38   fetch      Download objects and refs from another repository
39   pull       Fetch from and integrate with another repository or a local branch
40   push       Update remote refs along with associated objects
41
42 'git help -a' and 'git help -g' list available subcommands and some
43 concept guides. See 'git help <command>' or 'git help <concept>'
44 to read about a specific subcommand or concept.
45 See 'git help git' for an overview of the system.
```



#### 4.1.29 Test Case: t005

**Purpose:** This Test Case demonstrates embedded  $\text{\LaTeX}$  enumerate command.

1. Step #1 bzless check
2. Step #2 zip check

**Requirement:** AMBER-IUR-006

**Step:** 1

**Confirm:** Program bzless has been installed.

**Expectation:** bzless installation location is displayed.

**Command:** which bzless

**Execution start:** Jul 16, 2022 11:47:38.094802

**Execution end:** Jul 16, 2022 11:47:38.096561

**Test Result:** FAIL

**Evidence:** Starts on next line.

```
1 which: no bzless in (/home/traap/.local/share/nvim/lsp_servers/sumneko_lua/extension/  
server/bin:/home/traap/.local/share/nvim/lsp_servers/latex:/home/traap/.rENV/  
versions/3.0.3/bin:/home/traap/git/dotfiles/bin:/home/traap/.bin:/usr/local/sbin:/  
usr/local/bin:/usr/bin:/usr/bin/site_perl:/usr/bin/vendor_perl:/usr/bin/core_perl:/  
var/lib/snapd/snap/bin:/home/traap/.fzf/bin:/home/traap/bin)
```

**Step: 2**

**Confirm:** Program zip has been installed.

**Expectation:** zip installation location is displayed.

**Command:** which zip

**Execution start:** Jul 16, 2022 11:47:38.097364

**Execution end:** Jul 16, 2022 11:47:38.098874

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/zip
```

#### 4.1.30 Test Case: t006

**Purpose:** This Test Case demonstrates embedded  $\LaTeX$  enumerate command.

1. Step #1 curl check
2. Step #2 xxd check

**Requirement:** AMBER-IUR-006

**Step:** 1

**Confirm:** Program curl has been installed.

**Expectation:** curl installation location is displayed.

**Command:** which curl

**Execution start:** Jul 16, 2022 11:47:38.101067

**Execution end:** Jul 16, 2022 11:47:38.102543

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/curl
```

**Step: 2**

**Confirm:** Program xxd has been installed.

**Expectation:** xxd installation location is displayed.

**Command:** which xxd

**Execution start:** Jul 16, 2022 11:47:38.103275

**Execution end:** Jul 16, 2022 11:47:38.104747

**Test Result:** PASS

**Evidence:** Starts on next line.

```
1 /usr/bin/xxd
```

## 4.2 System Environment

The hyphen character replaced the underscore character, and the forward slash character replaced the backslash character throughout this section.

**ALLUSERSPROFILE:**

**APPDATA:**

**CLASSPATH:** See below.

1. nil

**COMPUTERNAME:**

**COMSPEC:**

**FP-NO-HOST-CHECK:**

**GIT-BRANCH:** master

**HOME:** /home/traap

**HOMEDRIVE:**

**HOMEPATH:** See below.

1. nil

**HOSTNAME:**

**JRE-HOME:**

**LANG:** en-US.UTF-8

**LOCALAPPDATA:**

**LOGONSERVER:**

**NUMBER-OF-PROCESSORS:**

**OLDPWD:** /home/traap

**OS:**

**PATH:** See below.

1. /home/traap/.local/share/nvim/lsp-servers/sumneko-lua/extension/server/bin
2. /home/traap/.local/share/nvim/lsp-servers/latex
3. /home/traap/.rbenv/versions/3.0.3/bin
4. /home/traap/git/dotfiles/bin
5. /home/traap/.bin
6. /usr/local/sbin
7. /usr/local/bin
8. /usr/bin
9. /usr/bin/site-perl
10. /usr/bin/vendor-perl
11. /usr/bin/core-perl
12. /var/lib/snapd/snap/bin
13. /home/traap/.fzf/bin
14. /home/traap/bin

**PRINTER:**

**PROCESSOR-ARCHITECTURE:****PROCESSOR-IDENTIFIER:****PROCESSOR-LEVEL:****PROCESSOR-REVISION:****PROFILEREAD:****ProgramData:****PROGRAMFILES:****ProgramFiles(x86):****ProgramW6432:****PSModulePath:** See below.

1. nil

**PUBLIC:****PWD:** /home/traap/git/amber**SESSIONNAME:****SHELL:** /bin/bash**SHLVL:** 2**SYSTEMDRIVE:****SYSTEMROOT:****TEMP:****TMP:****TZ:****USER:** traap**USERDNSDOMAIN:****USERDOMAIN:****USERNAME:****USERPROFILE:****WINDIR:**

## 5 Configuration Item Conclusion

This Report Package has satisfied the IUR for the Configuration Item described herein thus the Configuration Item is considered validated for its intended use.

### Change Summary

Change	Justification
[A] - Initial version.	New document.
[B] - Section 1 changes.	Reference gSOP.
[C] - Global report changes.	Amber is complaint autodoc rolling releases.