

GPD/STIP Test Plan

Version 2.3 (Public Release), June 2008

Table of Contents

[1. Introduction](#)

[1.1. Document Scope & Purpose](#)

[1.2. Target Audience](#)

[1.3. References](#)

[1.4. Copyright](#)

[1.5. Disclaimer](#)

[2. Goals & Methodology](#)

[3. Usage](#)

[4. Test Environment & Resources](#)

[5. Test Case Organization](#)

[6. Test Case Index](#)

1. Introduction

1.1. Document Scope & Purpose

The purpose of this document is to describe the tests that are necessary to prove that a GPD/STIP platform implementation complies with the GPD/STIP Specifications. It provides the basis for building suites of test software that prove GPD/STIP compliance. Such software suites are referred to as *GPD/STIP Test Suites* throughout this document.

1.2. Target Audience

The target audience for this document consists of the following groups:

- Providers of GPD/STIP platform implementations can use the test case descriptions to confirm they are interpreting the specifications correctly and as the basis for their own internal testing.
- Providers of GPD/STIP Test Suites can use it during development as a definition of all the test cases their test suites must cover.
- Users of GPD/STIP Test Suites may use it to understand the test cases that the test suite is executing.

Readers are assumed to be familiar with the GPD/STIP Specifications (see *References* below).

1.3. References

GPD/STIP Specifications v2.3. The specifications for version 2.3 of the GlobalPlatform Device (GPD) Small Terminal Interoperability Platform (STIP) is available for download from the GlobalPlatform web site at www.globalplatform.org. It consists of overview documents and full specifications of the GPD/STIP APIs. Familiarisation with these specifications is a prerequisite for readers of this document.

1.4. Copyright

Copyright © 2005-2008 GlobalPlatform Inc. All Rights Reserved.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights or other intellectual property rights of which they may be aware which might be infringed by the implementation of the specification set forth in this document, and to provide supporting documentation. The technology provided or described herein is subject to updates, revisions, and extensions by GlobalPlatform. Use of this information is governed by the GlobalPlatform license agreement and any use inconsistent with that agreement is strictly prohibited.

1.5. Disclaimer

The material in this specification is presented "as is", without any claims to its fitness for or applicability to a particular use. This document is subject to change without notice.

2. Goals & Methodology

The primary goal of this Test Plan is to provide a means to assess whether GPD/STIP platform implementations comply with the GPD/STIP Specifications. The methodology used to achieve this is:

- The Test Plan contains descriptions of test cases that aim to exercise all elements of the interfaces that are described in the GPD/STIP Specifications. The emphasis is on testing the *interfaces* provided by the service controls rather than the *functionality* of the underlying services. For example, the test for a good `SCSControl.powerOn()` operation checks that it results in a good `powerOnCompleted` event, but does not check that power has been applied to the card. It may be that the Test Plan will be extended at a later date to include functional testing.
- A *unit testing* approach is used. Thus the test cases for each method of each interface are specified in isolation (as much as possible). There is no attempt to specify test sequences that exercise all possible *combinations* of method calls.
- Indeed, the sequence in which test cases are performed is totally optional, because we want to specify only *what* must be tested. The test suite developer is at liberty to decide *how* the tests are performed.

At the current time, the scope of the Test Plan is restricted to the on-device testing of the GPD/STIP API specifications. This excludes the off-device activities of code generation and the building of STIP distribution files.

3. Usage

The Test Plan is designed to be used as follows:

1. Providers of GPD/STIP Test Suites must ensure their test suites cover all the test cases described in the Test Plan.
2. Providers of GPD/STIP platforms are responsible for having their platforms tested using a test suite that implements this Test Plan.
3. GlobalPlatform defines the rules of compliance assessment.

4. Test Environment & Resources

Details of the test environment for a particular test suite are left to the discretion of the test suite developer. However, the Test Plan authors make some assumptions:

- Test suites will consist of a number of stiplelets.
- Each stiplelet will perform a number of test cases.
- Wherever possible, the stiplelets will automatically check the results of each test case.
- Each stiplelet must report whether its tests passed or failed in such a way that these results can be saved as evidence of the test run and can be validated by a third party.
- Test suites must be able to handle platforms that do not implement all GPD/STIP features. For example, a platform may support only a subset of the cryptographic algorithms. Test suites could handle this in one of the following ways:
 - Execute all test cases and report precisely which method calls failed and what they resulted in, e.g. `AccessException(ACCESS_DENIED)` thrown by `CryptoControl.initSession()`, or `CryptoControl.initSessionCompleted()` with `status=EVT_ERROR` and `info=ERR_ACCESS_DENIED`. The test suite output would need to make the context of the failed calls clear, to facilitate validation that the failure is justified.
 - Allow configuration of the test suite to omit groups of tests, e.g. omit tests of the DSA algorithm. The test suite output would need to make it clear which tests were omitted. Note that some tests of the missing feature would still be necessary, to prove that use of it is denied in the correct way.

The resources required to test a particular platform can be generalised as follows:

- A validated test suite.
- The platform to be tested.
- One or more target devices on which to run the tests.
- Development tools for building the test stiplelets and downloading them to the target devices.
- The `smartcardslot` tests require a set of smart cards with particular characteristics. Details can be found in the `smartcardslot` Package Description.

5. Test Case Organization

The test cases are organized hierarchically. The levels of the hierarchy, from top to bottom, are: package, interface, method and test. The index in the next section below provides links to each package.

The documentation for each package consists of:

- Index - hyperlinks to the start of each interface and each method.
- Package Description - copied from the GPD/STIP Specifications. Some packages have some extra notes regarding the testing of the package.
- Interfaces - subsections for each interface belonging to the package.

The documentation for each interface consists of:

- Interface Description - copied from the GPD/STIP Specifications. Some interfaces have some extra notes regarding the testing of the interface.
- Methods - subsections for each method belonging to the interface.

The documentation for each method consists of:

- Method Description - copied from the GPD/STIP Specifications.
- Prototype - copied from the GPD/STIP Specifications.
- Parameters - list of names and descriptions, copied from the GPD/STIP Specifications.
- Additional Comments - some methods have a list of comments that apply to all the test cases that follow.
- Test Cases - a table of test cases for this method. For each test case there is:
 - TestRef - a unique reference number for the test.
 - Prerequisites - conditions necessary to set up the correct context for the test.
 - Parameters - every test case (except tests of status events and STIPML) consists of a single call to the method being tested. This column documents the parameter values to be passed on that call. Where a particular value is specified, that value must be used unless it is qualified by "e.g." or "for example".
 - Expected Results - the immediate results of the call, i.e. exception, return value or operation completion event.

The unique TestRef reference numbers have the following format:

```
packageID.interfaceID.methodID.testID
```

where:

- `packageID` is an alphanumeric mnemonic for the package, maximum length 4 characters, e.g. 'cryp' for the `stip.devicecontrol.crypto` package. A full list of these mnemonics can be found in the Test Case Index below.
- `interfaceID` is a numeric ID for the interface within the package. Because most packages contain at least a Control, an Operation Listener and an Operation Event, these interfaces are always numbered 1, 2 and 3 respectively.
- `methodID` is a numeric ID for the method within the interface.
- `testID` is a numeric ID for a test case for the method.

For example, the `open()` method of `stip.devicecontrol.crypto.CryptoControl` is allocated `methodID = 1` and its test cases are numbered:

- `cryp.1.1.1`
- `cryp.1.1.2`
- etc

Note that any particular test case will always keep the reference number that is first assigned to it. So if test cases are deleted (e.g. if a method is deleted from the specifications then all its tests will be deleted from the Test Plan), the other test cases will *not* be renumbered. There will just be a gap in the numbers in use from then on.

6. Test Case Index

Here are links to the test cases for each of the packages covered by this Test Plan:

packageID	Package/Link
acc	stip.access
cont	stip.control
util	stip.util
stip	stip.stiplet
stco	stip.stiplet.control
stmn	stip.stiplet.mngt
beep	stip.devicecontrol.beeper
ct	stip.devicecontrol.cardtransport
chv	stip.devicecontrol.chv
comm	stip.devicecontrol.comm
ser	stip.devicecontrol.comm.serial
mod	stip.devicecontrol.comm.serial.modem
cles	stip.devicecontrol.contactless
init	stip.devicecontrol.contactless.initiator
targ	stip.devicecontrol.contactless.target
cryp	stip.devicecontrol.crypto
x509	stip.devicecontrol.crypto.certificate.x509
date	stip.devicecontrol.date
file	stip.devicecontrol.file
led	stip.devicecontrol.led
msr	stip.devicecontrol.magstripereader
vol	stip.devicecontrol.mastervolume
med	stip.devicecontrol.media
net	stip.devicecontrol.net
dgrm	stip.devicecontrol.net.datagramsocket
http	stip.devicecontrol.net.http
https	stip.devicecontrol.net.httpserver
ssoc	stip.devicecontrol.net.serversocket
sock	stip.devicecontrol.net.socket
pow	stip.devicecontrol.power
siui	stip.devicecontrol.simpleui
scs	stip.devicecontrol.smartcardslot
spee	stip.devicecontrol.speech
tim	stip.devicecontrol.timer
ui	stip.devicecontrol.ui
brow	stip.devicecontrol.ui.browser
prin	stip.devicecontrol.ui.printer

vib	stip.devicecontrol.vibrator
xml	stip.devicecontrol.xml
ml	STIP_Markup_Language_(STIPML)