IDMS Test Plan

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev** | **Date** | **Author** | **Description** |
| 1.0 | 5/1/12 | Rob Silvernagel | Release Version |
| 1.6 | 5/21/13 | Jason Olmstead | Updated Restful endpoints, 1.6 new features and bug fixes. |

**Document Approvers & Sign-Off**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Approver** | **Role** | **Document Accept/Reject** |
| 5/1/12 | Mark Mecham | Synapse Test Manager | Accept |
| 6/1/13 | Mark Mecham | Synapse Test Manager |  |

**Table of Contents**

[1 Introduction and Purpose 5](#_Toc358731844)

[2 Background 6](#_Toc358731845)

[3 Scope 7](#_Toc358731846)

[This document explains the tests on the IDMS API, the IDMS databse tests, and performance metrics that must be met as well as the background and information necessary to perform these tests. 7](#_Toc358731847)

[4 Dependencies 7](#_Toc358731848)

[5 Ownership 7](#_Toc358731849)

[6 Related Documents 7](#_Toc358731850)

[7 System Overview 8](#_Toc358731851)

[7.1 IDMS Architecture 8](#_Toc358731852)

[7.2 IDMS ID Relationship 9](#_Toc358731853)

[7.3 Sequence Diagram 10](#_Toc358731854)

[8 Testing Approach 11](#_Toc358731855)

[8.1 Use Cases 11](#_Toc358731856)

[8.2 Test Strategies 11](#_Toc358731857)

[8.2.1 Test Data 11](#_Toc358731858)

[8.2.2 Test Execution 11](#_Toc358731859)

[8.2.3 Identifiers 12](#_Toc358731860)

[8.3 API Testing 12](#_Toc358731861)

[8.3.1 Test Case Structure 12](#_Toc358731862)

[8.4 Stress Testing 13](#_Toc358731863)

[8.5 Load Test 13](#_Toc358731864)

[9 RESTful Endpoints 14](#_Toc358731865)

[9.1 IDMS Endpoints 14](#_Toc358731866)

[9.1.1 Configuration 14](#_Toc358731867)

[9.1.2 Status 14](#_Toc358731868)

[9.1.3 Metrics 14](#_Toc358731869)

[9.1.4 Guest 14](#_Toc358731870)

[9.1.5 xBands 15](#_Toc358731871)

[9.1.6 Read-Only Configuration Test) 15](#_Toc358731872)

[10 Performance Targets 16](#_Toc358731873)

[11 Exit Criteria 17](#_Toc358731874)

[12 Release Changes 17](#_Toc358731875)

# Introduction and Purpose

The intent of the IDMS test plan is to provide a mid-level approach for testing IDMS and related services. The intended audience is anyone who may possibly be testing the component. From an agile “Kanban” approach this could be a QA team member, a developer, or anyone else with the appropriate technical expertise to interpret and execute the tests.

# Background

The ID Management System (IDMS) stores and maintains information about park guests, making it possible to link external systems with the xConnect components. IDMS utilizes a SQL Server database instance for data storage and a RESTful web service interface for data access. Although IDMS is not technically a System of Record (SOR) it does provide a systematic way of storing and retrieving guest information as well as looking up associations between guests, parties, xBands, and xConnect related IDs.

Many xConnect components rely on IDMS data for real-time operations. The accuracy and speed of the IDMS system is paramount to the overall performance of the xConnect system. From a quality assurance perspective, the priority for testing IDMS is: 1) Availability of RESTful endpoints, 2) Accuracy of the data, and 3) Response time.

# Scope

## This document explains the tests for the IDMS API, the IDMS database,, and performance metrics exit criteria as well as the background and information necessary to perform these tests.

# Dependencies

The ability for IDMS to function is dependent upon the following services.

| External | Description |
| --- | --- |
| xBMS | Guest creation and guest-xBand association |
| GxP SOR | gxp-link-id generation |

# Ownership

The contacts for each component reliant on IDMS are listed below.

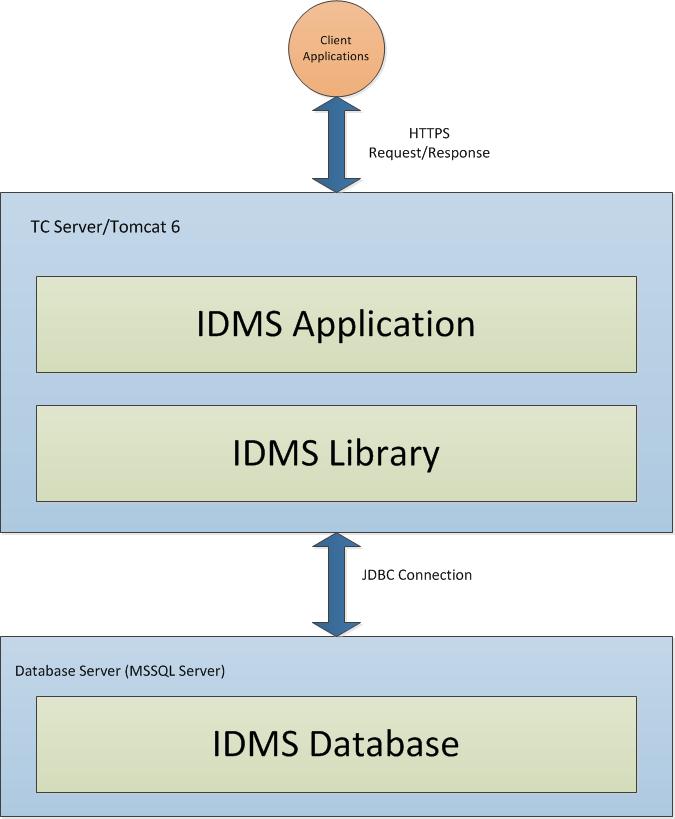
| Compo-nent | Developer |  | QA Tester |  |
| --- | --- | --- | --- | --- |
| IDMS | Robert Lantry | robert.lantry@synapse.com | Jason Olmstead | jason.olmstead@synapse.com |
| GxP | Scott Salley | scottsa@synapse.com | Robert Silvernagel | robs@synapse.com |
| xBRC | Arek Glabek | arkady.glabek@synapse.com | Michael Lampi | Michaella@synapse.com |
| xBRMS | Iwona Glabek | iwona.glabek@synapse.com | Francisco Gonzalez | francisco.gonzalez@synapse.com |

# Related Documents

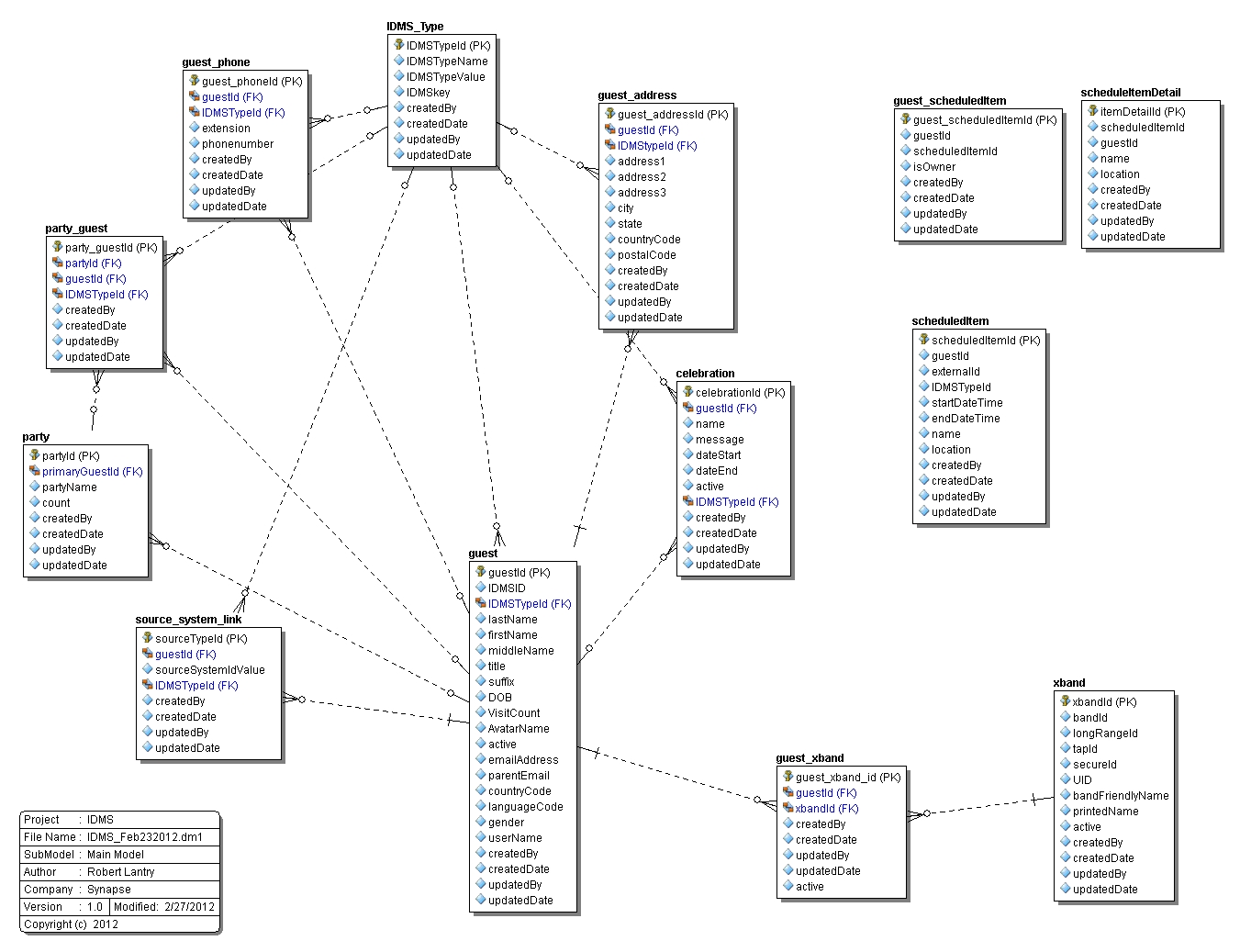
| Document Name | Relationship |
| --- | --- |
| 900-0056 IDMS ICD and Design | Defines the IDMS API |
| 900-0067 xConnect Test Plan | High level test approach to support xConnect software testing |
| 900-0075 xConnect System Troubleshooting | Troubleshooting guide for entire xConnect system including IDMS |
| 900-0136 xConnect IDMS Architecture | Describes system architecture for IDMS software as well as the network, hardware and software prerequisites |
| 900-000281 IDMS Test Cases | Excel document of the test cases and test description |

# System Overview

## IDMS Architecture



## IDMS ID Relationship



## Sequence Diagram

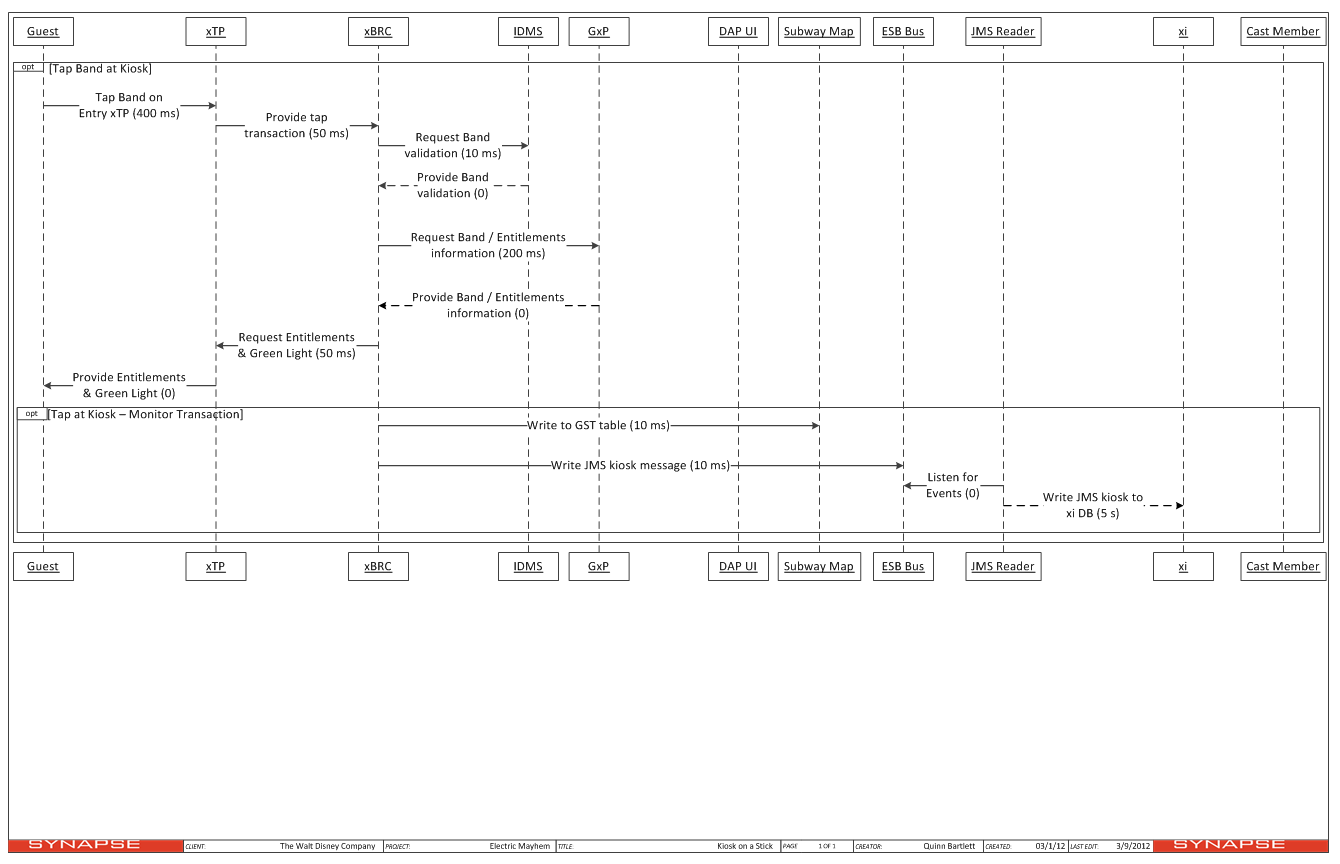


Figure 2 Sequence diagram for a kiosk tap, which is the same flow structure as a redemption tap.

# Testing Approach

One important aspect of testing IDMS is to ensure that the RESTful web services are available and functioning as intended. Another is validating that the database contains the appropriate data after any given set of operations are performed. This section outlines the approach to be used when testing these functions of IDMS.

## Use Cases

* A guest taps their card at a park kiosk to determine if their Space Mountain entitlement is valid at this time. The secure ID is read from the card and used to look up the guest in IDMS.
* A guest taps their band at park entry. An IDMS lookup identifies the guest by the secure ID before calling OMNI to verify park access.
* A guest’s xBand is detected by an xBR (long range reader) The xBRC uses the xBand’s UID to call IDMS and identify the guest.
* A guest sits down at a Great Food Fast table and the xBRC calls IDMS to identify the guest and look up their food order.

## Test Strategies

### Test Data

Test data defines the actual test cases that will be exercising IDMS to ensure proper functionality.

The test data will consist of five parts:

1. The hostname and port of the IDMS instance under test
2. The request, or the url of the RESTful endpoint being tested
3. The HTTP method (GET, POST, PUT, DELETE)
4. The request body, or payload of the request (if applicable)
5. The expected response

### Test Execution

IDMS tests are executed by making RESTful calls to the IDMS endpoints and validating the responses. Many of the IDMS endpoints use the ‘GET’ method. These endpoints can be tested from an internet browser such as Microsoft Internet Explorer or Mozilla FireFox. The format for the URL is:

http://{IDMS\_host}:{port}/IDMS/{RESTful endpoint}.

The response body will contain a JSON object which will be displayed in the browser.

While most calls can be easily tested from the browser, there are tools available to test the other methods such as ‘POST’, ‘PUT’, and ‘DELETE’. The recommended tool is ‘Fiddler’ which is available at www.fiddler2.com. Follow the same URL structure as previously mentioned. Select the appropriate method from the dropdown list. If the endpoint requires a request body, there is a text input box at the bottom of the Fiddler UI. After making a request, the request will appear in the ‘Web Session’ frame on the left side of the Fiddler UI. Select the request to see the response. The response headers will include an ‘HTTP 200’ to indicate success. Many times there will be a response body that also needs to be validated against the expected results.

### Identifiers

There are various ways of identifying a guest or xBand in the IDMS. The guestID serves as the primary key for a guest and uniquely identifies guests in the system. External systems have their own means of uniquely identifying guests. Tests that require a guest ID should also test the various external IDs in addition to the IDMS guest ID.

| External | ID Type | Example call |
| --- | --- | --- |
| GxP | gxp-link-id | http://localhost:8080/IDMS/guest/id;gxp-link-id=2468/xbands |
| XBMS | XBMS | http://localhost:8080/IDMS/guest/id;XBMS=314159/xbands |
| IDMS | xid | http://localhost:8080/IDMS/guest/id;xid=1357/xbands |
| IDMS | xband | http://localhost:8080/IDMS/guest/id;xband=98765/xbands |

## API Testing

The IDMS API consists of a set of RESTful endpoints. Manual testing of the API can be conducted using tools such as Fiddler or WireShark. Automated testing will be implemented via node.js or soapUI.

A list of the valid endpoints is contained in the ‘IDMS ICD and Design’. A terse reference is available in this document (see [RESTful Endpoints](#_RESTFUL_ENDPOINTS)).

### Test Case Structure

* The test data will consist of a request to IDMS as well as a HTTP method based on the specifications for the endpoint. The test data can be exercised either manually (via Fiddler or WireShark) or in an automated manner (using node.js scripts).
* The request will consist of the test data and the method
* The test result is the response that IDMS returns after receiving the request
* The test data should denote instances where the test results requires santization (i.e. removal of headers, decoding, etc)
* The sanitized test result will be compared to the expected test result and a pass/fail result will be generated.

## Stress Testing

Selenium Grid can be used to generate a high number of concurrent IDMS connections. The tests will be scalable and various performance metrics can be gathered with each load increase. The scalability will come from varying the number of Selenium remotes that are generating IDMS requests. Each remote will send IDMS a new request after the previous request’s response is received. Performance metrics will be used to determine how much latency is introduced as the load increases from 2->5->10->20->30->40 remotes/threads. Each test will run for a minimum of 5 minutes.

## Load Test

The load tests will tie in with the system load tests such as ‘day in the life of the park’ tests. These tests verify that IDMS is providing the needed endpoints and data to fulfill requests. A detailed list of the load tests can be found in the ‘900-000281 IDMS Test Cases’ document.

# RESTful Endpoints

IDMS employs an API or a set of RESTful endpoints that allow clients to access/interact with IDMS data. The base URL for IDMS is: http://{IDMS\_HOST}:{PORT}/IDMS. A more complete reference can be found in the ‘IDMS ICD and Design’ document. The endpoints listed below are the ones that are supported at the time this document was last edited.

## IDMS Endpoints

### Configuration

| Key | Description | HTTP Action |
| --- | --- | --- |
| /configuration | The XView of configuration properties. | GET |

### Status

| Key | Description | HTTP Action |
| --- | --- | --- |
| /status | The current IDMS status. | GET |

### Metrics

| Key | Description | HTTP Action |
| --- | --- | --- |
| /metrics | The metrics on the performance of the calls made. | GET |

### Guest

| Key | Description | HTTP Action |
| --- | --- | --- |
| /guest/{guestId}/xbands | Returns a list of xBands assigned to the guest. | GET |
| /guest/locators | List of identifier types currently supported by IDMS | GET |
| /guest/{guestId}/identifiers | Returns a GuestIdentifierCollection of GuestIdentifier objects | GET |
| /guest/{guestId} | Returns guest details | GET |
| /guests/id/{guestId} | Returns guest details | GET |
| /guests/{guestId} | Returns guest details | GET |
| /guest/{guestId}/online-profile | Returns online profile of guest | GET |
| /guest/{guestId}/profile | Returns online profile of guest | GET |
| /guest/search/{searchName} | Returns a list of guests that have a last name, first name, or email address. | GET |

### xBands

| Key | Description | HTTP Action |
| --- | --- | --- |
| /xband/bandid/{bandId} | Returns an XViewXband object by BandId. | GET |
| /xband/lrid/{longRangeId} | Returns an XViewXband object by longRangeId. | GET |
| /xband/tapid/{tapId} | Returns an XViewXband object by tapId. | GET |
| /xband/secureid/{secureid} | Returns an XViewXband object by secureId. | GET |
| /xband/uid/{uid} | Returns an XViewXband object by uniqueId. | GET |
| /xband/public/{publicid} | Returns an XViewXband object by publicId. | GET |
| /xbands/id/{xbandId} | Returns an XViewXband object by xBandId. | GET |
| /xbands/{xbandId} | Returns an XViewXband object by xBandId. | GET |
| /xbands/lrid/{longRangeId} | Returns an XViewXband object by longRangeId. | GET |
| /xbands/bandid/{bandid} | Returns an XViewXband object by bandId. | GET |
| /xbands/secureid/{secure} | Returns an XViewXband object by secureId | GET |

### 

### Read-Only Configuration Test

To test whether or not the read-only configuration is behaving as expected, each IDMS instance must be tested by attempting to update records. If the instance is read-only, the PUT or POST should fail and the data in that isntance’s database should be unchanged. If the instance is writable, the PUT or POST should succeed and the database should be updated with the new value. See ‘900-000281 IDMS Test Cases’ document for specific tests.

| Key | Body Format | Description | HTTP Action |
| --- | --- | --- | --- |
| /xband/ | JSON | Adds an xband to the IDMS | POST |
| /xband/{xbandid} /xband/id/{xbandid} /xband/bandid/{bandid} /xband/lrid/{lrid} /xband/tapid/{tapid} /xband/secureid/{secureid} /xband/uid/{uid} /xband/public/{publicid} | JSON | Updates the details of an xband using a specified id. | PUT |

# Performance Targets

IDMS response times will vary based on a number of factors, including network traffic, external components, and the endpoint being requested. In order to ensure that IDMS meets these performance targets a full xConnect system should be exercised under a reasonable load for at least 15 minutes. This includes at least one xBRC connected to an xTP with at least 10 redemption calls to GxP per minute. At the end of this test the IDMS performance metrics can be viewed by opening a browser and browsing to the URL: http://{IDMS host}:8080/IDMS/metrics. Ensure that the average response times fall below these thresholds:

| **Endpoint** | **Maximum Acceptable Average Response Time (in ms)** | **Critical for Redemption?** |
| --- | --- | --- |
| GetXBandsByIdentifier | 5 | Yes |
| GetGuestByEmail | 5 | No |
| GetGuestProfileById | 5 | No |
| GetGuestIdentifiers | 5 | Yes |
| GetXbandBySecureId | 5 | Yes |

# Exit Criteria

1. BVT tests are all passing with the exception of known and acknowledged failures that are documented in the release notes
2. SIT tests are all passing with the exception of known and acknowledged failures that are documented in the release notes
3. Performance targets are met

# Release Changes

There were a number of changes to to the API to not allow certain methods to be called on an endpoint when the configuration is set to read-only. The calls that used to be able to be made will no longer work when the configuration is set to read-only. These tests cases are listed in the ‘900-000281 IDMS Test Cases’ document. Below is a list of changes and fixed bugs in this release:

* Removed the “meta” API
* Removed “celebrations’ API
* Removed “itinerary-items” API
* Removed “import” API
* Removed “party” API
* Removed “resort-reservations” API
* Removed “visit” API
* Added the feature to check read-only status
* Changes to the way the IDMS is configured
* Retrieving celebrations not working (fixed)
* Returned data not returning match documentation (fixed)
* Database scripts not checking previous version before updating (fixed)
* Guest creation does not support swid (fixed)
* Import xbands reponse not reporting 201 (fixed)
* XbandId not resolving to guest (fixed)
* IDMS version not being set correctly (fixed)
* IDM does not return guests name in result when querying xband (fixed)
* Saving guest identifier fails (fixed)
* IDMS should show which database it expects (fixed)
* 200 reponse when adding guest to party (fixed)
* Guest name update fails when processing messages from xBMS (fixed)
* Database read error on xband lookup (fixed)
* Get xView Guest failed (fixed)
* Incorrect swid values being returned by /xband endpoint (fixed)
* Calling guest search endpoint causes exception (fixed)
* IDMS guest-data endpoint returns 500 (fixed)
* Guest Name update fails when processing messages from xBMS (fixed)
* xBand records not being created from XBMS.XBAND messages (fixed)
* Incorrect value for swid returned by /xband enpoint (fixed)
* Calling guest search endpoint causes exception (fixed)