



## Janvi Soft QA Process Whitepaper

JanviSoft  
3910 Riverbend Ter, Fremont, CA 94555  
Phone: 510-475-9999, Fax: 510-892-4444

## TABLE OF CONTENTS

1.0 Approach.....	3
1.1 Overview.....	3
1.2 Definition of Scope.....	3
1.3 Tests to be Conducted.....	3
1.4 Tools Utilized.....	5
1.5 Project Inception Checklist.....	5
2.0 Test Plans.....	6
2.1 Test Plan Creation Process.....	6
2.2 Test Plan Structure.....	7
2.3 Sample Test Plan.....	9
3.0 Test Plan Execution.....	10
3.1 Manual Execution Process.....	11
3.2 Reporting Details.....	12
3.3 Handling Procedures.....	13
3.4 Severity Levels.....	14
4.0 Fix Validation Process.....	15
5.0 Reporting.....	17
5.1 Daily Summary Data.....	17
5.2 Weekly Summary Report.....	18
5.3 End of Cycle Reports.....	18
6.0 Compatibility Test.....	19
6.1 Browser Checks.....	19
6.2 Operating Systems.....	19
7.0 Automated Testing.....	19
8.0 Stress, Load and Capacity Testing.....	19
8.1 Load Testing.....	20
8.2 Stress Testing.....	20
8.3 Capacity Testing.....	20
8.4 Reporting Terms.....	20
9.0 Minimal Acceptance Tests.....	21
10.0 Communication.....	21

## 1.0 Approach

### 1.1 Overview

JanviSoft provides offshore and on-site quality assurance services. We have a dedicated staff of QA professionals to perform test plan creation, script development for automated testing, manual and automated test plan execution, and stress, load and capacity testing.

The following pages outline our overall, fundamental approach to QA. While each project varies we recommend, and utilize the following procedures outlined in this document.

### 1.2 Definition of Scope

A definition of scope document is created to describe the QA phase and schedule for each release. The goal of the scope document is to achieve agreement and commitment from all departments involved to the outlined conditions and timelines identified. The scope document contains:

#### Areas to be tested

- New System Modules
- Modified System Modules
- Unchanged System Modules

#### Areas not to be tested

- System Module – Reason  
(i.e. Customer Service – Internal application, Unchanged)

#### Environment Requirements

- Connection thru: [VPN, Firewall Access, DNS]
- DB Access Required & Provided [Y/N]
- Platforms Needed [Win 95, Win 98, Win NT, Win ME, Win 2000, Solaris, Mac 8.6, Mac 9, Mac 10]
- Browser Types Needed [IE 4.0, IE 4.1, IE 5, IE 5.5, NS 4.08, NS 4.75, NS 6.0]

### 1.3 Tests to be Conducted

After identifying the areas of the application to be tested, the type of tests must be defined. The below type of tests related to Black Box testing are considered.

#### Functional Tests

A set of functional tests will developed based on client documentation, and review of any existing systems. Dedicated QA Engineers will execute these functional tests manually.

#### Compatibility Tests

JanviSoft QA lab provides for testing to be conducted on multiple browsers and operating systems. The desired testing combinations and process (including priority of testing) will be determined during the scope definition process.

#### Stress & Performance Tests

We use market-leading tools to assess how the site/product holds up under different loads and we provide the customer with extensive data to help scale the site/product up to meet higher loads.

### Automated Testing

JanviSoft has the capacity to devote resource(s) dedicated to developing automated test scripts. The areas of the application where automated testing are applied will be identified by JanviSoft and the client during the scoping phase. Typically, areas which remain relatively constant and that are considered reasonably stable are considered strong candidates for automated testing.

### 1.4 Project Inception Checklist

JanviSoft will manage the below checklist, which tracks the information transfer concerning product documentation, testing documentation and tools, application access and application knowledge transfer.

Client:				
Project Name:				
Main Area	Item	Onus	Date	Info
Bug Tracking	Web Based System Available to All Parties			
	Client Users Created			
	Users Created			
Documentation	Functional Requirements Document(s) to QA	Client		
	Users Manual to QA (if available)	Client		
	Old Test Plans to QA (if available)	Client		
Testing Environment	URL for QA Environment	Client		
	URL for Staging Environment	Client		
	System User Names/Passwords	Client		
	QA Lab Resources Allocated (OS/Browser etc)			
Communication	Product Walkthrough	Client		
	Bug Handling Procedures	Client /		
	Emergency Contact Information	Client /		

	General Contact Information (recipients of daily status reports)	Client		
Schedule	Scheduled Delivery of First Build	Client		

## 2.0 Test Plans

### 2.1 Test Plan Creation Process

JanviSoft takes considerable care in developing and managing test plans and test cases. Developing accurate, clear and concise test cases is critical to the success of black box, functional testing.

#### Client Documentation

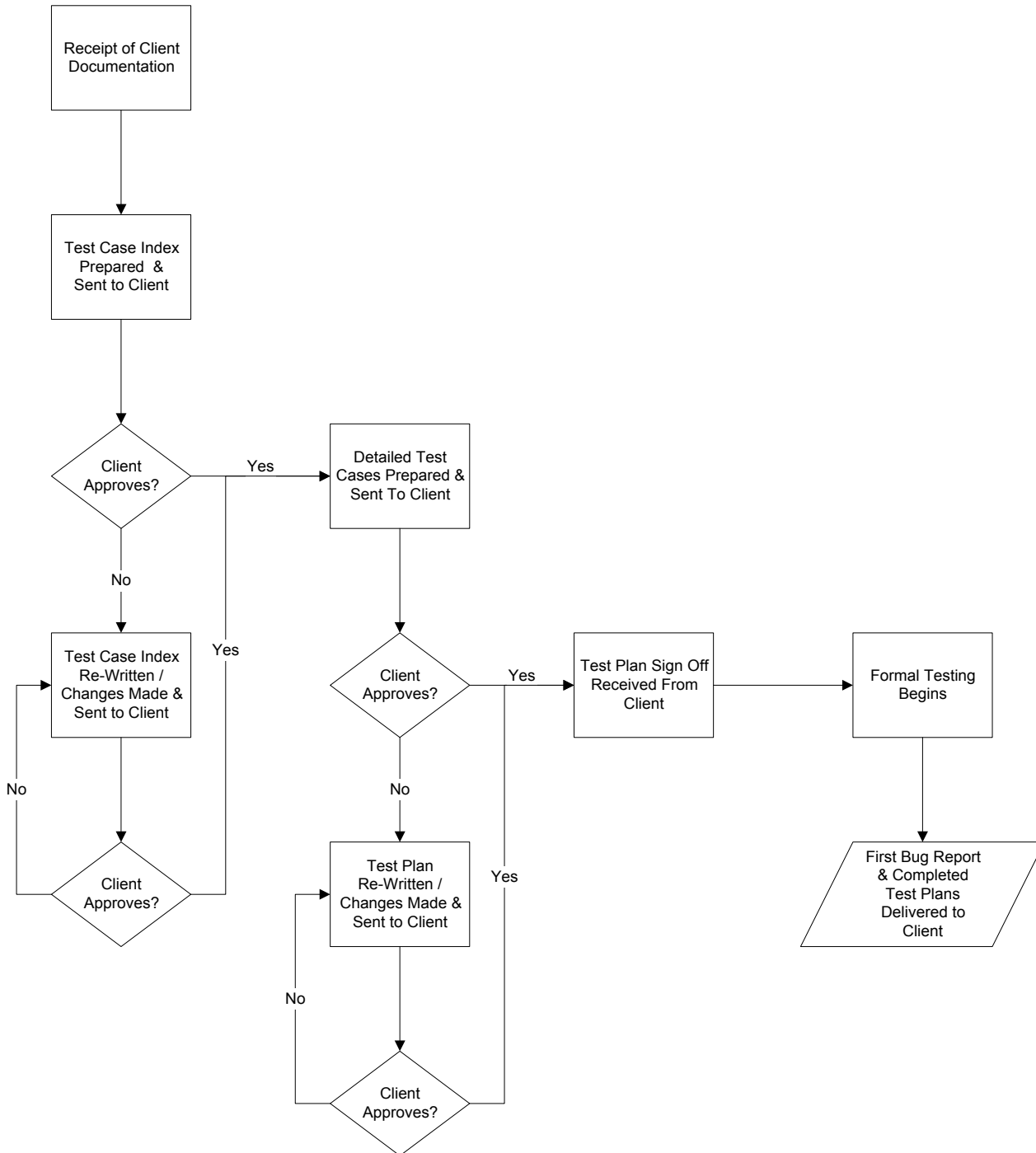
The first step in developing test cases is receiving and reviewing the Functional Requirements Documents provided by the client. The FRD review process includes: of FRDs:

- Unit Level Functionality dissected
- Understanding of each pages' rendering logic
- Creation of Application flow
- Integration Scenarios

#### Test Plan Sign-off Process

JanviSoft follows the below diagram in order to gain client approval of the test cases developed by JanviSoft. The basic tenets of the sign-off process include:

- Developing Test Plan Indexes:
- Developing Detailed Test Scenarios (test cases):
- Review and Adjustment cycle with client:
- Test Plans accepted



**Test Plan Version Integrity**

In order to maintain the integrity of the testing, JanviSoft places the accepted version placed as read only in specified file location. Any modifications/additions to test plan during execution are sent to client and maintained in 'working' version of test plan.

**2.2 Test Plan Structure**

JanviSoft follows a strict structure for our test plans. The test plan structure is designed to provide quick reporting and to assist in problem tracking.

Front Page

Each test plan contains a first page, which contains the following for each section and for the entire test plan: # Test Cases, # Cases Completed, # Passed, # Failed, # Blocked, % Completed, % Passed, % Failed, % Blocked.

TOTAL TESTS		TOTAL PERCENT	
# Tests:	<b>540</b>		
# Completed:	<b>242</b>	% Complete:	<b>45%</b>
# Passed:	155	% Passed:	29%
# Failed:	87	% Failed:	16%
# To be executed	<b>298</b>	% To be executed	<b>55%</b>
# Blocked:	12	% Blocked:	2%

Column Headings

Each detailed test scenario (test case) has the following column headings: Test Case ID, Status, Defect #, Test Case Area, Test Steps, Expected Results

Test Case id	Status	Defect #	Test Area Reference Id	Test Steps	Expected Result
--------------	--------	----------	------------------------	------------	-----------------

Test Areas

All JanviSoft test plans are divided into sections. Sections consist of groupings of test scenarios. Examples of individual test plan sections include: Registration, Login, User Rights & Privileges,

Section	#Tests	% Complete	% Passed	%Failed	%Blocked
1. Login	11	100%	100%	0%	0%
2. Deployment Set	20	100%	95%	5%	0%
3. Views	52	98%	98%	2%	2%
4. Edit from Package view	19	100%	100%	0%	0%
5. Edit from File view	15	100%	67%	33%	0%
6. Edit from Host view	13	100%	100%	0%	0%
7. File Menu Options	23	96%	0%	0%	4%
8. Other Options	4	100%	100%	0%	0%
9. Package Settings	19	100%	89%	11%	0%
10. File Settings	28	100%	89%	11%	0%
11. Host Settings	27	100%	100%	0%	0%
12. Log	20	100%	100%	0%	0%
13. Deploy	35	100%	100%	0%	0%
14. Server Configurations	57	100%	49%	51%	0%
15. Administration Configurations	81	85%	58%	42%	15%
16. Scripts	15	100%	67%	33%	0%
17. Help	77	100%	91%	9%	0%
18. Adhoc Testing	24	100%	29%	71%	0%

## 2.3 Sample Test Plan

JanviSoft test plans are divided into two components, the test plan index and the collection of detailed test scenarios.

### Test Plan Index

The test plan index is developed to provide a high level overview of the application sections and the type of tests to be performed. The test plan index acts as a table of contents for the final collection of test scenarios.

A sample of one section of a test plan index is provided.

#### **5 Edit Deployment set option for object**

- 5.1 Add File to deployment set
  - 5.1.1 Add One Available File (With host and Package)
  - 5.1.2 Add One Available File (with package only)
  - 5.1.3 Add All Available File
  - 5.1.4 Create New file and Add
- 5.2 Delete File from deployment set
  - 5.2.1 Delete Existing File
  - 5.2.2 Delete Newly Added File
- 5.3 Add Package to deployment set
  - 5.3.1 Add One Available Package
  - 5.3.2 Add All Available packages
- 5.4 Delete Package from deployment set
  - 5.4.1 Delete Existing Package
  - 5.4.2 Delete Newly Added Package
- 5.5 Add Host to deployment set
  - 5.5.1 Add One Available Host
  - 5.5.2 Add All Available Hosts
  - 5.5.3 Create New Host and Add
- 5.6 Delete Host from deployment set
  - 5.6.1 Delete Existing Host
  - 5.6.2 Delete Newly Added Host

### Detailed Test Cases

Our detailed test scenarios are developed so each QA engineer will follow the exact pre-defined steps in order to isolate and test specific components of the application. As testing is conducted, new test scenarios may be added. These cases are added in their appropriate section and once developed are completed for each testing cycle.

One detailed test scenario is shown below.

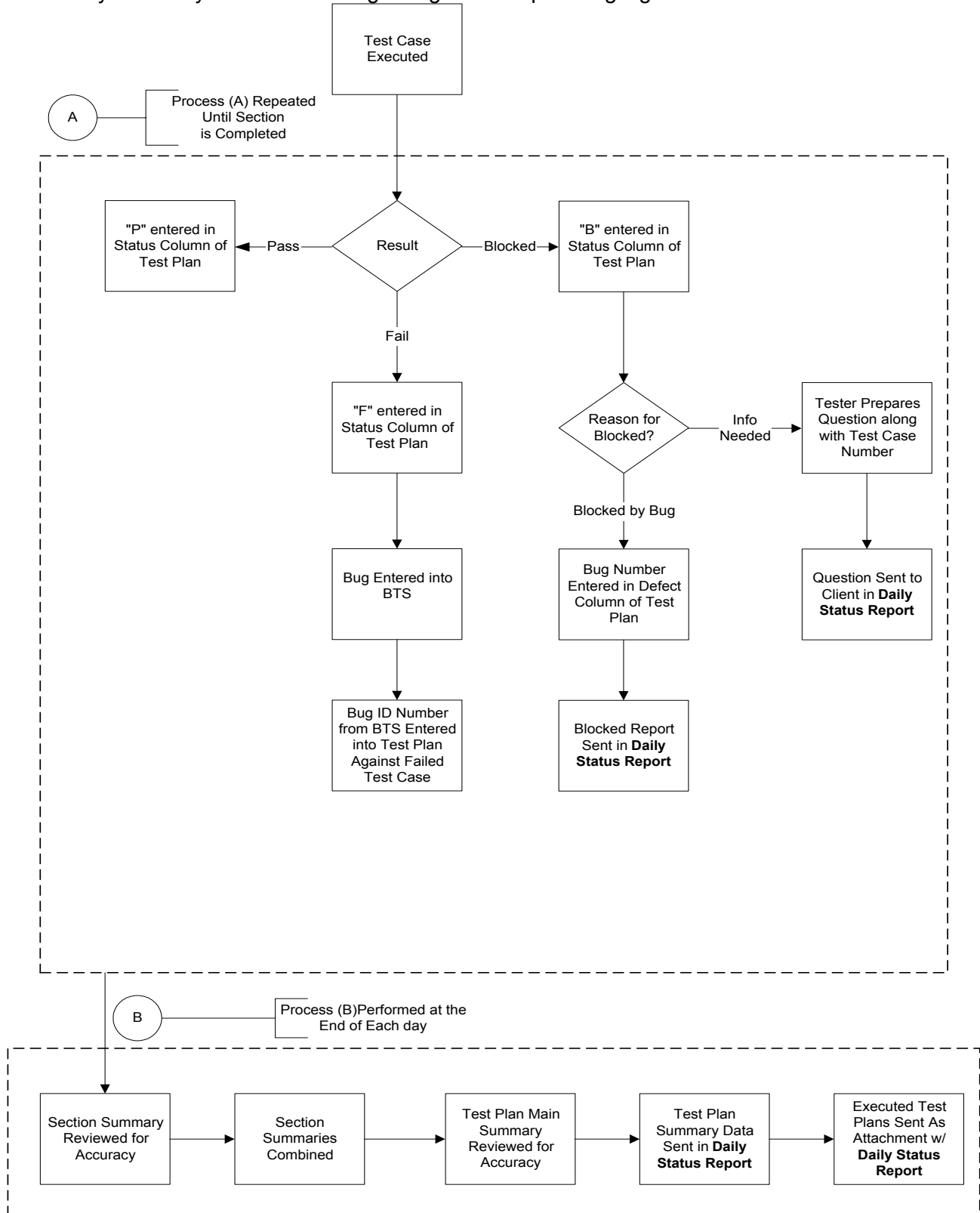
Test Case id	Status	Defect #	Test Area Reference Id	Test Steps	Expected Result
5.1.1			<b>File - Add - with host and with Package</b>	<ol style="list-style-type: none"> <li>1. Launch the Kelly application</li> <li>2. Login with the Valid User</li> <li>3. Select a Deployment Set Icon to be opened</li> <li>4. From the Main Menu, Select the Files View Tab</li> <li>5. Select the Deployment Set Radio button</li> <li>6. Click the Add Link</li> <li>7. In the Select Objects Screen, Select a File (having Packages and Hosts) in the Available List Box</li> <li>8. Click &gt; Button, then Click the Submit Files Button</li> <li><b>9. Verify this (Result: Selected File should be listed under the Deployment Set in the Main Menu Screen of the File view)</b></li> <li>10. Click on the Package View tab</li> <li><b>11. Verify this (Result: Selected File should be listed below the Packages under the Deployment Set in the Main Menu Screen of the Files view)</b></li> <li>12. Click on the Hosts View tab</li> <li><b>13. Verify this (Result: Selected File should be listed below the Hosts under the Deployment Set in the Main Menu Screen of the Hosts view)</b></li> <li>14. In the Main Menu Screen, click the log link</li> <li>15. Select Date of Creation from Date Dropdown; username from the user Dropdown</li> <li>16. Click the Apply Filter button</li> <li><b>17. Verify this (Result: Information for the adding of this File should be displayed)</b></li> <li>18. Click on Return to main Screen</li> <li>19. In the Main Menu Screen, Click on deploy link</li> <li>20. select the option "full"</li> <li>21. click Proceed to Deployment button</li> <li>22. Verify this (Result: Deployment request Sent Screen should be displayed and then the Log screen should be displayed)</li> <li>23. From the Log Screen, Select the Date of deployment from Date dropdown; Username from the User Dropdown; Deployment from the Action Dropdown</li> <li><b>24. Verify this (Result: Information Regarding this Deployment should be displayed)</b></li> </ol>	<ol style="list-style-type: none"> <li>1. Check the File view (Step 9)</li> <li>2. Check the Package View (Step 11)</li> <li>3. Check the Host View (Step 13)</li> <li>4. Check the Log file for the added File (Step 17)</li> <li>5. Check the Log file for the deployment status (step 24)</li> </ol>

### 3.0 Test Plan Execution

JanviSoft QA engineers follow a strict process, while performing manual testing. The execution of test cases involves the following steps.

### 3.1 Manual Execution Process

JanviSoft QA Engineers follow the process shown below in executing test cases and in compiling end of day summary data. Details regarding each step are highlighted below.



- Execute scenarios from top to bottom on assigned browser/platform
- Enter Pass, Fail, Blocked or Question in Results Column
  - If Fail, open bug tracker
    - Enter steps performed into bug tracker & write description
    - Repeat steps to verify bug occurs before submitting
    - Perform steps on other platform/browser combos per pre-arrangement with client
    - Add Additional Notes to Bug Tracker, Submit
    - Write bug number in "Default ID" column
    - Upon completion of each section, fill in section details
  - If Blocked
    - Enter the default number of issue which is blocking test case
    - If no specific issue exists, open issue, record number and enter in result for blocked issue
  - If Question
    - Enter a detailed doubt in the result column
    - Before the end of *each day*, consult JanviSoft QA manager
    - If the issue cannot be resolved and scenario tested, a detailed *Clarification Request* will be sent to client concerning that issue.
- Each days test plans will be saved in a dated folder
- All bug numbers will be carried over from previous days testing and placed in the default number so that multiple bugs are not opened on the same issue
- When bugs are closed, the closed date will be added underneath the default number.
- Before execution of a Test Plan begins. The previous day's test plan is modified all data in "Status" column are removed. ALL data in "Defect" column remains. Upon completion of 1 Full Cycle (all cases executed or blocked), the daily executed plans are combined for an 'End of Cycle" report.

### 3.2 Reporting Details

JanviSoft placing a strong emphasis on reporting identified issues correctly and thoroughly. In order to provide our clients with the best possible information concerning an issue, we take the following steps while reporting quality assurance issues.

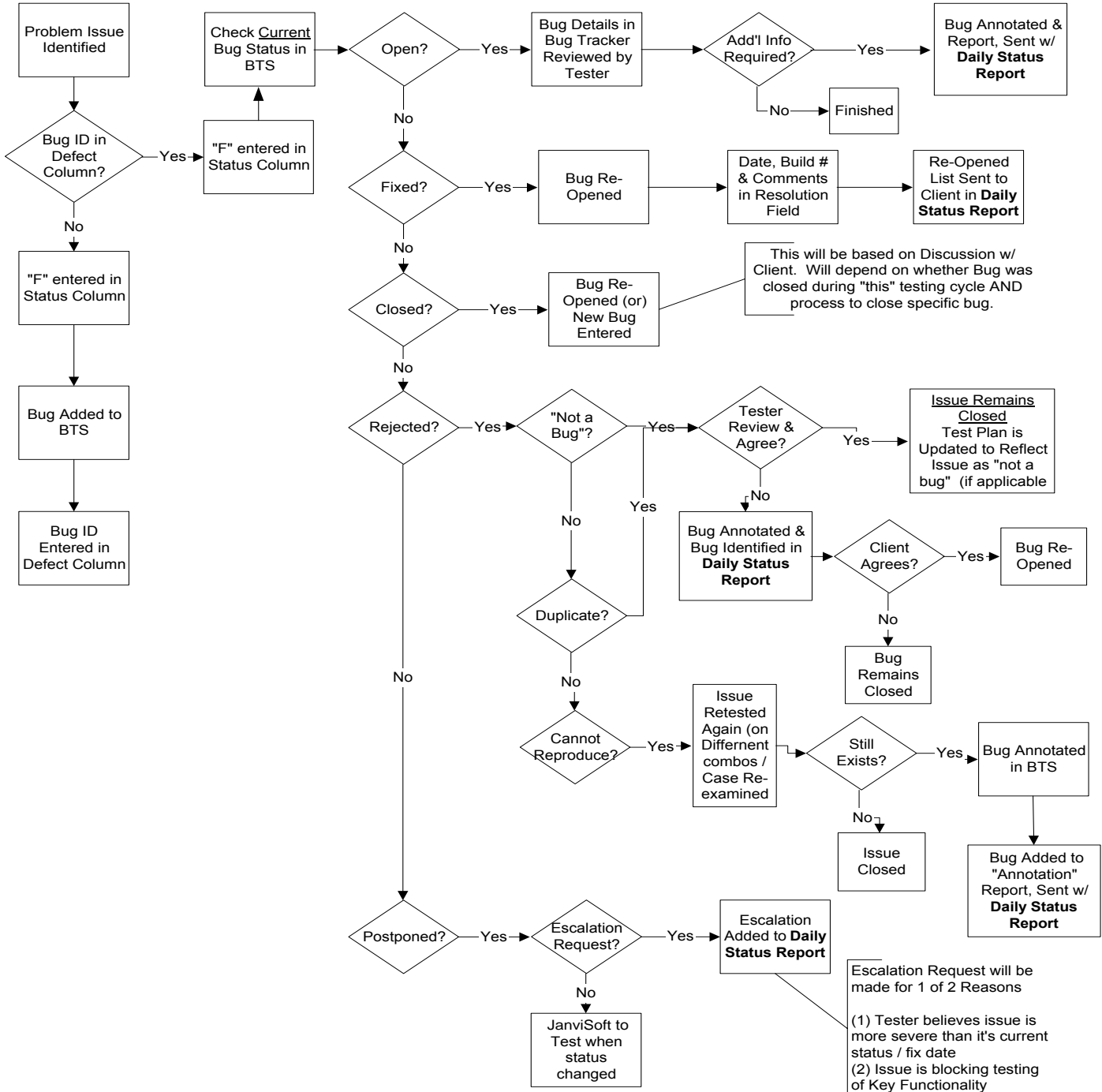
- Identify: Following Test cases & logical thinking
  - Test cases provide a *basis* for testing
  - Other scenarios should be run as application grows; these should be added to the test cases
- Investigate: After the initial bug is found
  - Investigate opportunities to eliminate variables
  - Identify associated pages or areas where the immediate bug shows impact
  - Perform 'gray' box investigating by accessing the database (if applicable)
  - Rework the bug until its impact is fully understood.
- Reporting: JanviSoft QA engineers follow a strict reporting process. This includes the reporting the following for each item.
  - Date Reported
  - Reported By
  - Version Reported IN (build number)
  - Environment (which section of code stream - QA, INT, DEV, Production)
  - Title
  - Area Definition (section name of the application)
  - Platform(s)

- Browser(s)
- Type (bug, enhancement)
- Severity (as reported by QA engineer)
- Detailed Description – Elements (with date/personnel stamp)
  - Login: provide user name, password, access rights, group etc..
  - Landing Page: Define page you are taken to.
  - Navigation: Define link/selection area & link/selection chosen
  - Navigation: Define landing page, describe page (continue 3 & 4 as required)
  - Issue: Identify the outcome of the sequence
  - Issue Clarification: Identify the expected results
  - Issue Exposure: Identify how this impacts every area - other screens, database etc.
  - Issue Wrap-up: Specific Details eliminating variables or providing developers insight into problem
  - Testing Wrap-up: when this was last tested, when it last worked, how it last worked etc.
  - Final Emphasis: Specific Environments, Specific User Groups, anything previously stated which needs more emphasis
  - Screen shots where/when applicable

### **3.3 Handling Procedures**

In order to accurately identify, investigate and report problem issues over the course of multiple testing cycles, JanviSoft executes a thorough process once a problem issues has been identified. This process is shown in the below diagram.

The below flow is used when a problem issue is found, while executing the test plans. The diagram is not a flow designed to review fixed bugs. That flow is outlined in the "Fix Verification" section of this document.



**3.4 Severity Levels**

A severity level is supplied for each problem issue identified. JanviSoft identifies five levels of severity.

Critical to show stopper: Critical  
Program aborts and the system cannot function further

Severity Level 2: High

Major functionality of the application is performing improperly.

Severity Level 3: Medium

A minor or non-mission critical aspect of the application is performing improperly. The existence of this error does not affect the users ability to access all crucial areas and functionality of the application.

Severity Level 4: Low

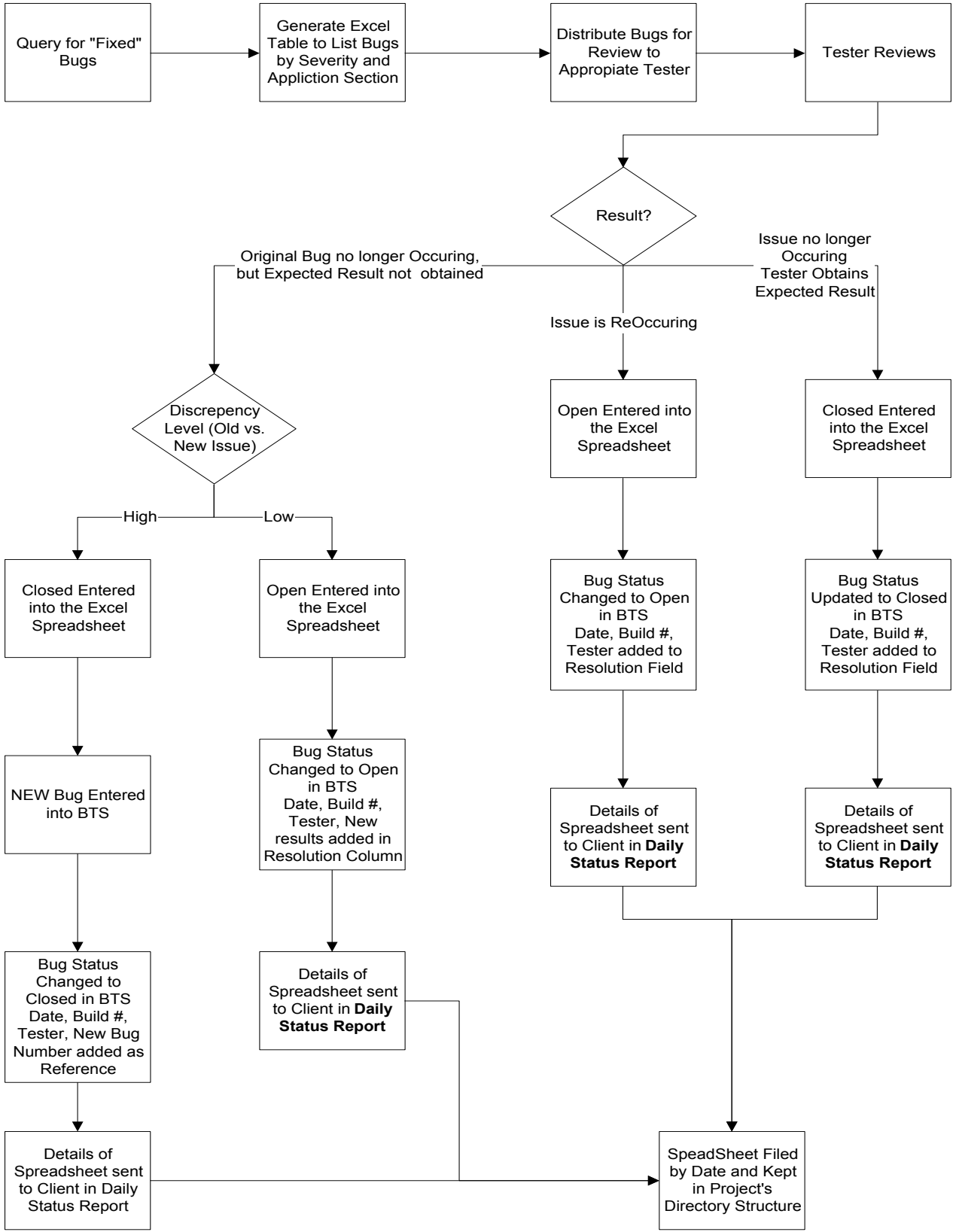
There are cosmetic or minor usability issues. The application functions properly even with the existence of these issues.

Severity Level 5: Change

These are change suggestions provided by JanviSoft QA engineers. The application still meets its functional and design requirements.

#### **4.0 Fix Validation Process**

The fix validation process is a critical component to success software development. To that end, JanviSoft relies on a series of steps so issues reported by development as “fixed” are properly handled. We retest each of these items, annotation item details within the bug tracker, and change the issue’s status accordingly.



## 5.0 Reporting

JanviSoft provides daily, weekly and end of cycle reporting throughout the entire QA cycle.

### 5.1 Daily Summary Data

In addition to entering new bugs and closing fixed bug in the bug tracking application, JanviSoft provides our clients the following detailed information.

- Number of test cases completed
- Number of new issues identified and reported
- Number of test cases, which failed
- Number of test cases blocked
- The executed test plans
- The current list of existing bugs by severity
- The test-tracking document. A sample test tracking document is show

Test Tracking Document

Date	Build ID	Section	Browser	New tests against this Build	Tests repeated against this Build	Total tests run this day	Tests Passed	Tests Failed	Tests Blocked	New Issues	Existing Issues
9-Jul	1	UI	IE 4.01	178	0	178	115	63	28		
10-Jul	1	UI	IE 4.01	154	0	154	94	60	63		
11-Jul	1	BE	NA	22	0	22	8	14	11		
12-Jul	1	CL	NA	24	0	24	17	7	37		
13-Jul	1	BE	NA	33	0	33	14	19	31		
16-Jul	1	UI	NS 6.01	185	0	185	135	50	37		
17-Jul	1	UI	NS 6.01	174	0	174	110	64	34		
18-Jul	1	UI	IE 5.0	229	0	229	146	83	26		
19-Jul	2	UI	IE 4.01	230	0	230	196	34	21		
20-Jul	2	UI	IE 4.01	266	0	266	178	88	21	3	39
23-Jul	2	BE / CL	NA	70	0	70	56	14	23	6	0
24-Jul	2	BE/CL	NA	53	0	53	50	3	1	1	2
25-Jul	2	BE/CL	NA	31	0	31	30	1	0	0	1
26-Jul	2	UI	NS 4.08	275	0	275	243	32	4	0	12

## 5.2 End of Cycle Reports

When the JanviSoft QA engineers execute all test cases for a build, we supply our clients a complete set of test results for the cycle. We provide the following information:

- Each completed test plan
- List of bugs found in this testing cycle
- List of bugs fixed in this testing cycle
- List of bugs deferred to a later release
- Full test tracking document for the completed cycle

## 6.0 Compatibility Test

JanviSoft performs functional testing on a variety of platform/browser combinations. The testing combinations will be identified during the scope process.

### 6.1 Browser Checks

JanviSoft can perform cross-browser compatibility tests to ensure applications can perform properly on different browsers. We will adjust the machine configurations within our QA lab to meet our client's application browser requirements.

#### Sample Browser List

3.	I.E. 5.5
6.	N.S 6.0
7.	N.S. 6.1

### 6.2 Operating Systems

JanviSoft maintains multiple operating systems on which to perform testing. Based on the client's requirements we can perform the functional testing on these operating systems in various combinations with the browsers above.

1	Linux
2	Solaris 2.6, 5.6, 7.2, 8.0
3	Various Windows Versions including: Win 95, Win 98, Win NT, Win ME, Win 2000
4	Multiple MAC Versions including Mac 8.6, Mac 9, Mac 10

## 7.0 Automated Testing

JanviSoft utilizes Mercury WinRunner to develop automated test scripts. Scripts are developed and updated by JanviSoft. The depth and breath of utilizing automated testing will be set at the beginning of the project. As the QA project progresses and the product becomes more stable, there are additional areas where automated testing may be beneficial. JanviSoft and our clients maintain a constant dialog concerning the potential deployment of automated testing.

## 8.0 Stress, Load and Capacity Testing

JanviSoft utilizes Mercury Interactive's LoadRunner to conduct stress, load and capacity tests. The details of our approach can be found in the JanviSoft QA Stress, Load and Capacity Testing document. The below provides an overview of our approach.

### 8.1 Load Testing

Testing an application against a requested number of users. The *objective* is to determine whether the site can sustain this requested number of users with acceptable response times.

### 8.2 Stress Testing

Load testing over an extended period of time. The *objective* is to validate an application's stability and reliability.

### 8.3 Capacity Testing

Testing to determine the maximum number of concurrent users an application can manage. The *objective* is to benchmark the maximum loads of concurrent users a site can sustain before experiencing system failure.

### 8.4 Reporting Terms

Report information containing the following information is provided to or clients.

#### Load size

This is the number of concurrent Virtual Clients trying to access the site.

#### Throughput

The average number of bytes per second transmitted from the ABT (Application being tested) to the Virtual Clients running this Agenda during the last reporting interval

#### Round Time

It is the average time it took the virtual clients to finish one complete iteration of the agenda during the last reporting interval.

#### Transaction Time

The time it takes to complete a successful HTTP request, in seconds. (Each request for each gif, jpeg, html file, etc. is a single transaction.) The time of a transaction is the sum of the Connect Time, Send Time, Response Time, and Process Time.

#### Connect Time

The Time it takes for a Virtual client to connect to the Application Being Tested.

#### Send Time

The time it takes the Virtual Clients to write an HTTP request to the ABT (Application being tested), in seconds.

#### Response Time

The time it takes the ABT (Application being tested) to send the object of an HTTP request back to a Virtual Client, in seconds. In other words, the time from the end of the HTTP request until the Virtual Client has received the complete item it requested.

#### Process Time

The time it takes WebLoad to parse an HTTP response from the ABT (Application being tested) and then populate the document-object model (the DOM), in seconds.

#### Wait Time (Average Latency)

The time it takes from when a request is sent until the first byte is received.

#### Receive Time

This is the elapsed time between receiving the first byte and the last byte.

### **9.0 Minimal Acceptance Tests**

Once the entire testing cycle has been completed and the application is being considered to be moved out of the QA environment, a final set of pre-defined tests should be conducted. This will be a sub-set of the detailed test scenarios developed from each area of the test plan. The results of the acceptance test will determine whether the application has passed the QA exit criteria and is ready to be deployed to a staging or live environment.

### **10.0 Communication**

Ongoing communication is conducted between our dedicated QA team and our clients. The communications mechanisms employed by our QA unit are listed below.

- Daily Emails
- Weekly Conference Calls
- Weekly yahoo or other online chat sessions
- Weekly status reports containing entire weeks completed work and projections for upcoming week
- End of Cycle Reports
- Any additional recommendations/requirements suggested by client.