Blaise Testing

Margaret Tang & Daniel Collison, Statistics Canada

1. Introduction

Statistics Canada (STC) has used Blaise to develop survey collection instruments since 1999. Currently, there are more than 200 Statistics Canada surveys in production. This includes monthly, quarterly, and annual surveys across the agricultural, business, and social areas for head office and the six regional offices across Canada.

In Statistics Canada, each survey typically uses the Blaise build[1] that was chosen at the start of the cycle. Once the survey is in production, the development managers are reluctant to change to another version of Blaise for future cycles. Survey managers only change to a new Blaise build if there are new features/fixes required for the next production cycle.

Different surveys have different components and utilize different features of the Blaise system. To minimize testing efforts and to lower the risk of discovering costly issues in later stages, it is necessary to establish a standard test strategy for the core features of the Blaise system used by most Statistics Canada surveys.

2. Blaise Testing Complexity

A Blaise survey instrument includes multiple questions and complex routing rules. Each survey instrument can have user defined elements such as data types, edits/checks for question fields, outcome codes and multiple interviewing languages. Moreover, different Blaise surveys can also have customizable features such as menu items to interact with external applications or utilize external databases. When all of these factors are taken into account, the Blaise testing complexity increases significantly.

Due to time constraints, Blaise survey testing concentrates primarily on the Data Entry Program for survey content and behavior. However, the Blaise system consists of many components including survey development, batch processing, data entry, call scheduling, data storage and maintenance (e.g. Hospital and DayBatch). Different versions of Blaise introduce new and modified features in different components that may not always work with existing survey instruments. Problems may arise with the Call Scheduler or Hospital as they are not part of the Blaise survey testing focus. To handle the complexity of testing these Blaise components, it becomes essential to establish a testing process to validate a Blaise build before recommending it for development.

3. Blaise Build Testing Goals

Statistics Canada has chosen to take advantage of the release of Blaise 4.8 to establish a standardized and reusable test process that systematically evaluates a new Blaise build. This will be accomplished by assessing the compatibility of the new build against current production needs, validating its ability to co-exist with other builds, and determining the changes required within a survey for it to be able to adopt the new Blaise build. The

[1] Blaise build – a specific version or release of the Blaise survey processing system

second goal is to establish a set of acceptance criteria that a Blaise build must satisfy before it can be recommended for survey development use. The third goal is to provide an estimate for time and effort needed to migrate to the new build. And, the final goal is to investigate the new features of a build and make recommendations on their applicability for STC surveys.

4. Blaise Build Testing Process

The Blaise build testing process has been established using the Rational Unified Process approach. This process considers the test effort from the perspectives of roles, activities, and artefacts.

4.1 Blaise Build Testing Roles

4.1.1 Test Manager

This role leads the overall test effort. This includes quality and test advocacy, resource planning and management, and resolution of issues that impede the test effort.

4.1.2 Test Designer

This role defines the test approach and ensures its successful implementation. This includes identifying the appropriate techniques, tools and guidelines to implement the required tests, and providing guidance on the corresponding test resources requirements.

4.1.3 Test Analyst

This role identifies and defines the required tests, monitors detailed testing progress and results in each test cycle, and evaluates the overall quality.

4.1.4 Tester

This role conducts tests and logs the outcomes of his/her testing.

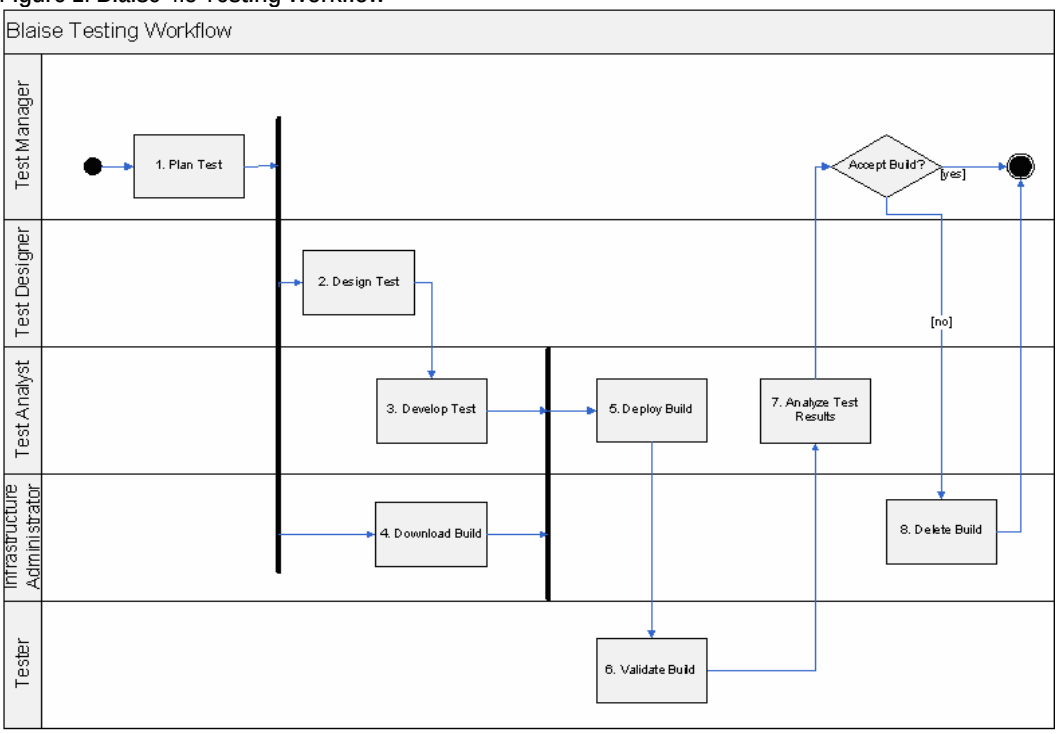
4.2 Blaise Build Testing Activities

The objective of the Blaise Testing Workflow is the verification that the Blaise under review will satisfy the common usage scenario and the acceptance criteria for Blaise applications in Statistics Canada. This involves the following steps:

- Establishing the scope and focus of the test
- Designing and implementing the required tests
- Performing the tests and reporting the problems discovered

Different testing roles are responsible for different activities in the workflow. As time progress, manual tests will be gradually enhanced or replaced by automated tools to improve high testing quality and/or lower testing cost.

Figure 1. Blaise 4.8 Testing Workflow



4.3 Blaise Build Testing Artefacts

4.3.1 Master Test Plan

This artefact defines the goals and scope of the test project, the items being targeted, the approach to be taken, the resources required and the deliverables to be produced.

4.3.2 Test Case

This artefact defines a set of test inputs, execution conditions, and expected results for the purpose of making an evaluation of some particular aspect of a target test item.

4.3.3 Test Bed

This is an environment created for Blaise build testing purposes. It includes commonly used Blaise and non-Blaise applications in STC.

4.3.4 Test Script and Procedure

This artefact defines steps to be executed automatically (script) or manually (procedure).

4.3.5 Test Log

This artefact provides a detailed status of the test effort and notes.

4.3.6 Test Evaluation Summary

This artefact presents a summary and analysis of the test results for review and assessment. It may also give recommendations on how to adapt the Blaise build.

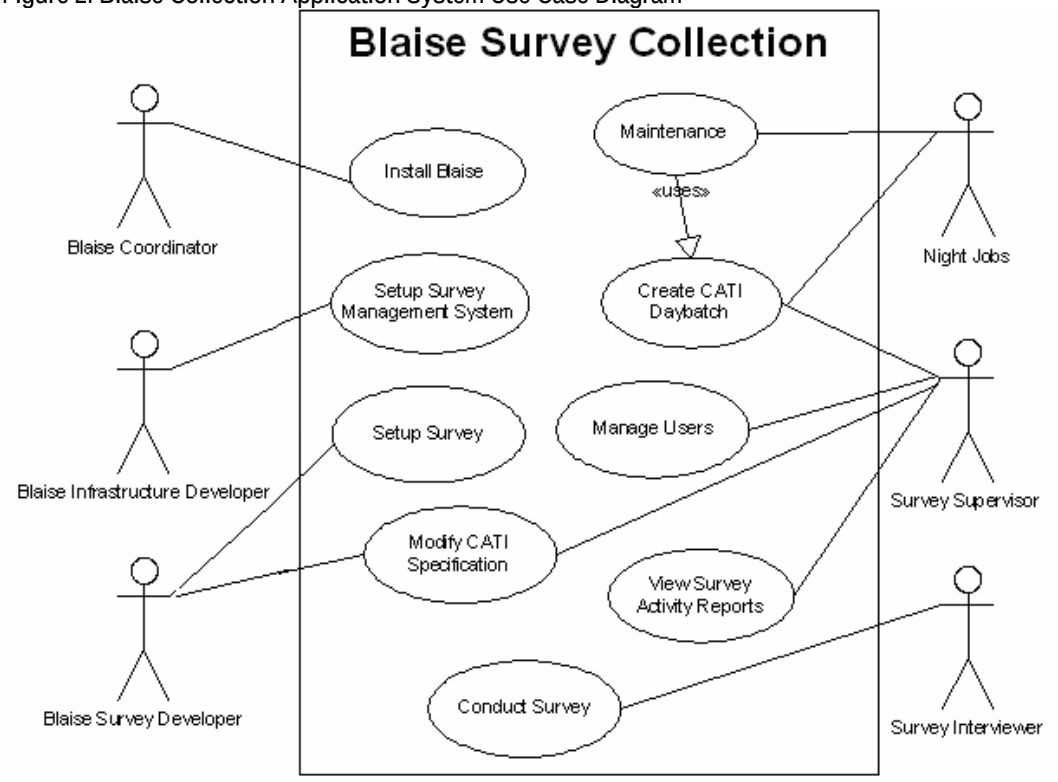
Table 1. Summary of Test Activities and Artefacts

	Activities	Artefacts
Test Manager	Plan Test	Master Test Plan
Test Designer	Design Test	Test Cases, Test Bed
Test Analyst	Develop Test	Test Scripts/Procedures
	Analyze Test Results	Test Evaluation Summary
Tester	Execute Test	Test Logs

5. Blaise Build Testing Approach

The Test Manager first identifies the scope and focus of the test in the Master Test Plan.

Figure 2. Blaise Collection Application System Use Case Diagram



The Test Designer then creates a suite of Test Cases and a Test Bed that will cover the scope. For example, for the Install Blaise Use Case, there will be one Test Case for installing the Blaise core system and another Test Case for installing the Blaise services.

One Use Case can generate multiple Test Cases based on functionality or options. Test Cases can also cover other aspects such as usability, reliability, performance and supportability (including compatibility and existing infrastructure constraints)

Based on the Test Cases, the Test Analyst will create Test Scripts or detailed Test Procedures for the Tester. For example, to simulate survey case load, Blaise Emulator scripts will be used to establish multiple Emulator sessions.

Next, the Tester will execute Test Scripts and record the results in the Test Logs so the Test Analyst can create the Test Evaluation Summary for a Blaise build. Finally, the Test Manager will accept or fail the build based on the Test Evaluation Summary.

The cycle restarts with the Test Analyst modifying Test Scripts/Procedures to reflect the peculiarities of the next build that is received. If the test scope is changed, the Test Designer will create new Test Cases and the Test Analyst will create the necessary Test Scripts/Procedures for the Tester.

6. Blaise 4.8 Build Testing Experience

There are 3 main focus areas in the Blaise 4.8 Build Testing:

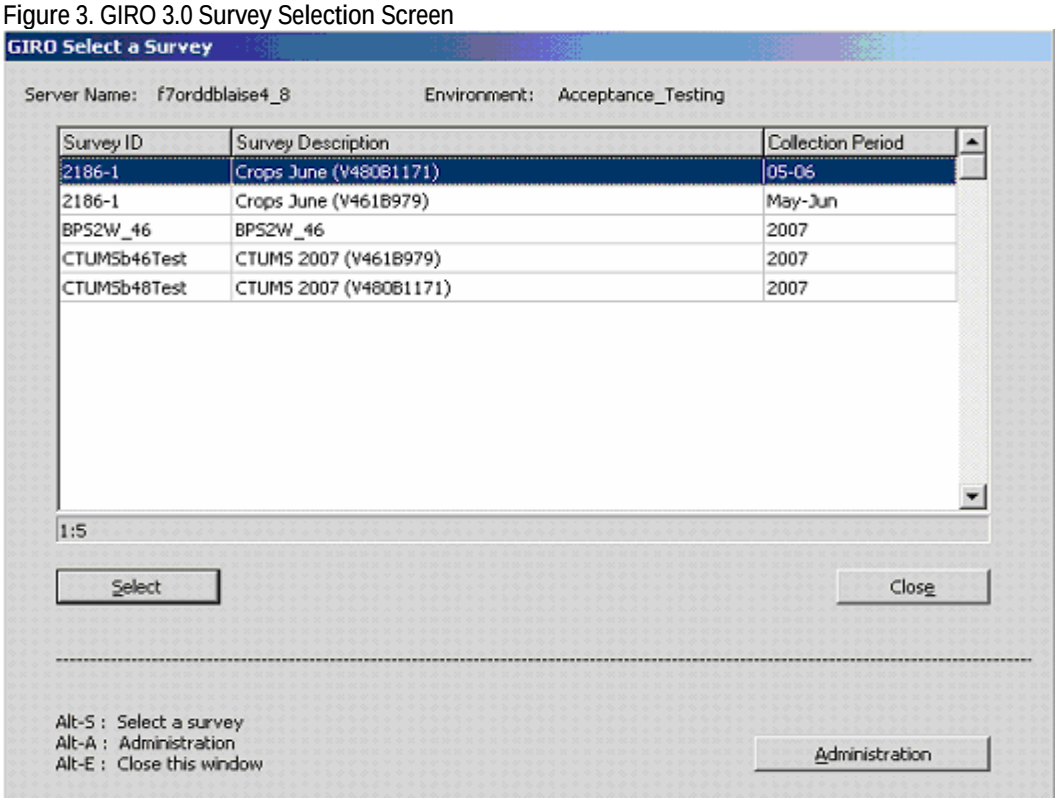
- Blaise Server Architecture: Blaise Database Service and Blaise CATI Service
- Backward Compatibility: The support of common survey collection activities
- New Features: Blaise DataLink, Blaise CATI events and Maniplus Interchange

6.1 Blaise 4.8 Build Testing Test Bed

The Blaise 4.8 Build Testing Test Bed consists of a Survey Management System, a Test Survey and several Survey Collection Applications.

The Survey Management System used is GIRO 3.0 (General Interface for Regional Offices), developed using Blaise 4.6 (build 979). It handles Survey Management and User Management functionalities for surveys in Statistics Canada's regional offices.

Figure 3. GIRO 3.0 Survey Selection Screen



The Test Survey used is CTUMS 2007 (Canadian Tobacco Use Monitoring Survey). It is developed using Blaise 4.6 (build 979). A 4.6 copy is used as the benchmark and its source code is recompiled in 4.8 and setup for Blaise 4.8 Testing.

The Survey Collection Applications included in the Test Bed are VB Browser (Survey Data Browser) and CATI Specification Regional Interface. They are both Visual Basic applications that interact with Blaise using the BCP (Blaise Component Pack).

Survey Activity Reports are developed using SAS 9.1 with Excel. Manipula scripts and Delphi DLLs used for Data Extraction complete the Test Bed.

6.2 Hardware Resources

Table 2. Summary of System Resources

	Operating System	Comments
Blaise Testing Server	Windows 2003	1 Virtual Server
Development Server	Windows XP	1 PC used for configuration test
Developer PC	Windows XP	2 developer PCs
Client Testing PC	Windows XP	10 PC from Testing Lab (as needed)

6.3 Blaise 4.8 Server Architecture

When testing the Blaise 4.8 Services, the Blaise 4.8 Test Survey is used for Data Entry and the Blaise Emulator is used to simulate a multi-user working environment. Memory and CPU usage of the server are monitored using Spotlight for Windows. Network Traffic is monitored by STC's Infrastructure Management team.

Table 3. Summary of Test Status

Version	Status	Notes
4.80.1129	Failed	Slow Blaise Database Service Performance for Batch Jobs
4.80.1146	Failed	Data Entry Program unable to work with CATI Service
4.80.1159	Failed	Slow/Unstable CATI Specification when modifying large file
		Slow WAN performance when all items are in Head Office

6.4 Backward Compatibility

Blaise 4.8 is deployed on the Blaise Testing Server in the same fashion as Blaise 4.6. The Test Bed is used to evaluate the common features used in STC. It also investigates the steps needed to upgrade to Blaise 4.8.

Table 4. Test Items and Blaise Features Covered

Test Item	Function	Blaise Feature
GIRO	User Manager – Update CATI	Blaise CATI Specification Program
	Select/Browse	Blaise API (VB 6 Application)
	CATI Specification	Blaise CATI API (VB 6 Application)
	Reports	Interaction with SAS and Excel
CTUMS2007	Compile	Blaise Parser (Command Line)
	Load	Manipula
	Functions Menu	Maniplus
	Interview	Data Entry Program
Maintenance	Check Data Storage Integrity	Blaise Hospital Program
	Data Extraction	Manipula
	Day Batch Creation	Blaise CATI Management Program
	BTH Extraction	Blaise Transaction History files
	Report Generation	Manipula and SAS
	Audit Trail Extraction	Audit Trail (Blaise Audit.dll)

Table 5. Summary of Test Status

Version	Status	Notes
4.80.1171	Suspended	Suspended due to news of API changes in build 1180
4.80.1180	Suspended	Suspended due to availability of build 1190
4.80.1190		First Production Blaise 4.8 Build

Table 6. Summary of Common Issues

Component	Issue	Notes
Blaise Parser	Parser Rules Tightened Up	Code is checked more rigorously
CATI Specification	New Format (.btr)	Existing .bts will need to be migrated to the new format
CATI Management	Microsoft XML 4.0 required	All users will need to install the Microsoft XML 4.0 locally

6.5 Other Blaise Features

Individual Test Scripts are created for Blaise DataLink, Blaise CATI Events, Alien procedures and Maniplus Interchange. These are used to evaluate possible future survey development and potential deployment/maintenance issues in user environments.

7. Conclusion

Blaise 4.8 Build Testing is a long process. Establishing a standard test suite helps reduce testing effort and improves testing quality. It also facilitates early error detection and problem reporting. As time goes on, more features will be added to the standard test suite. With automation, more extensive testing can be done in much less time. Eventually, the cost and risk of upgrading to a more recent Blaise build will be much lower and will encourage survey development managers to explore the full potential of Blaise.

8. References

IBM Corporation, Rational Unified Process Version 7.0, 2005