Test Documentation

S7 Dev team

June 1, 2012

Part I
Testing plan

1 Introduction

The main objective with this test plan is to determine the robustness of the applications core functionality.

In consultation with the client we have decided to test only the core functionality of the application. This decision has been made in regard to the remaining time to deadline, where we find other tasks to have greater priority.

2 Relationship to other documents

To better understand the scope of the application and this test plan, we recommend that the reader read subreport four before proceeding. After reading subreport four, the reader will have a clear overview regarding the functional requirements, non-functional requirements, technical design and purpose of the application.

3 Features to be tested/not to be tested

All functionality in the model handlers will be tested. For Employees, Departments and Projects, this include the following methods:

- Insert()
- Retrieve()
- RetrieveAll()
- DeleteAll()

- UpdateStatus()[1]

The functionality in the data fetcher class (`FetchUpdates.cs`) and the two XML parser classes (`FullFeedParser.cs` and `StatusParser.cs`) will also be roughly tested.

As explained in the introduction, only core functionality is tested. Therefore this testplan does not include tests regarding functionality contained in the `Config.cs` class. Also it has been decided that actual GUI testing fall outside the scope of this test plan.

4 Approach

Where it is possible we will use the NUnit testing framework. NUnit is an open source unit testing framework for Microsoft .NET.

NUnit enables us to isolate each part of the program we want to test and show that the individual parts work correct. By testing the parts of a program first and then testing the program itself, integration testing becomes much easier.

Our NUnit testing has three main purposes:

- Verification
- Validation

The verification process confirms that the software meets its technical specifications. A specification is a description of a function in terms of a measurable output value given a specific input value under specific preconditions. The validation process confirms that the software meets the business requirements. A defect is a variance between the expected and actual result.

Where unit testing are not feasible, we use manual testing to achieve verification and validation, and thus that our code works as expected.

5 Test cases

[1] Only for Employees