

VI. SYSTEM TESTING

Testing Objectives

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

VII. CONCLUSION

This paper studies the problem of pulling out co-locations from spatially uncertain data with probability intervals. It has defined the possible world model with prospect intervals, and proves that probability intervals of all possible worlds are feasible. It has also defined the interrelated concepts of probabilistic widespread co-locations. Then it has proved the closure possessions of prevalence point probability

REFERENCES:

- [1] C.C. Aggarwal et al., "Frequent Pattern Mining with Uncertain Data," Proc. 15th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, pp. 29-37, 2009.
- [2] P. Agrawal, O. Benjelloun, A. Das Sarma, C. Hayworth, S. Nabar, T. Sugihara, and J. Widom, "Trio: A System for Data, Uncertainty, and Lineage," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 1151-1154, 2006.
- [3] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 487-499, 1994.

- [4] C.C. Aggarwal and P.S. Yu, "A Survey of Uncertain Data Algorithms and Applications," IEEE Trans. Knowledge and Data Eng. (TKDE), vol. 21, no. 5, pp. 609-623, May 2009.
- [5] T. Bernecker, H-P Kriegel, M. Renz, F. Verhein, and A. Zuefle, "Probabilistic Frequent Itemset Mining in Uncertain Databases," Proc. 15th ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD '09), pp. 119-127, 2009.
- [6] C.-K. Chui, B. Kao, and E. Hung, "Mining Frequent Itemsets from Uncertain Data," Proc. 11th Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD), pp. 47-58, 2007.
- [7] C.-K. Chui and B. Kao, "A Decremental Approach for Mining Frequent Itemsets from Uncertain Data," Proc. 12th Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD), pp. 64-75, 2008.
- [8] M. Ester, H.-P. Kriegel, and J. Sander, "Knowledge Discovery in Spatial Databases," Proc. 23rd German Conf. Artificial Intelligence (KI '99), (Invited Paper), vol. 1701, pp. 61-74, 1999.