DEVELOPING AN INTELLIGENT GENERATOR FOR SEMI-ACTUAL TEST DATA

S. A. Hameed, A. M. A. Al-Abbasi

Centre of Measurement and Evaluation, University of Bahrain,Bahrin
shihab@iiu.edu.my, amajid@iiu.edu.my

Abstract: The actual test data generation is one of the difficult and expensive parts of applying software-testing techniques. Many of the current test data generators suffer from the reduction of user's confidence in generated test data and testing process. This is because of focusing on developer and database administrator viewpoints regardless of users concerns and focusing on data type and structure regardless of meaning. This paper proposes a model of an intelligent generator for semi-actual test data with the aim of increasing users confidence in software testing. The model uses samples of real data as a resource data and a set of efficient generation techniques based on statistical methods such as permutations, combination, sampling, and statistical distributions. The selection of the suitable structure and generation technique is based on one of the intelligent soft computing techniques such as fuzzy logic, neural network, heuristic, or genetic algorithm. The generated test data is validated according to the data specifications then tested by one of the normality testing techniques to be close to the real world or environment of the testing processes. This model offers the ability of simulating real environments.

Keywords: Software Testing, Test Data Generation, Semi-Actual Data, Intelligent Generator, Simulation.

1. INTRODUCTION

During the 1990's, the primary challenge and goal of software engineering was the production of quality software and the reduction cost of computer-based solutions that can be implemented with software [1, 2]. To improve software quality, software testing is one of the essential tools. It is one of the complicated problems in the life cycle of software development, which is expensive (40-50% of the total software development cost) and labor intensive [3, 4]. Software is now being applied in critical situations to control valuable machinery, handle money, and safeguard human lives. The failure in such situations can be disastrous; therefore there is much need for efficient software testing to reduce the risk of software [5, 6]. Software testing requires set(s) of test data. The automation and improvement of test data generation will reduce the cost of software development and testing. Unfortunately, automatic test data generation still faces many problems. These problems can be summarized as follows:

- Shortage and inefficiency of some test data generation techniques or tools.
- Duplicated or conflicting descriptions of the same or similar data items used in applications that have similar goals.
- The generated data holds high ratio of meaningless items, which may not reflect the specifications, culture, and environment of the population under test.
- Using inefficient set of generation technique(s) based mainly on random number generator (RND) or similar functions.
- Focusing on developer or database administrator viewpoints regardless of user concerns.
- Minimum participation of the user in test data generation process.
- Lack of user's confidence in the generated test data, testing process, and consequently in application under test.

To overcome these problems this paper proposes a model of an intelligent generator for semi-actual test data. It is an improvement on a previous work [7]. The proposed model uses intelligent, soft computing, approaches such as fuzzy logic, neural network, heuristic, or genetic algorithm [8] to provide approximate solutions to selected problems. These approaches are suitable for selecting the suitable data item's structures and generation technique for the proposed model. The generated test data is then checked according to the normality test(s) to satisfy the required specifications.

2. GOAL AND OBJECTIVES

The main goal is to generate suitable set(s) of semi-actual test data to support the software testing process, and to improve user confidence in testing software applications. The function of this generator to:

- Generating different volumes, types, and structures of test data.
- Generate different sets of test data that contains high ratio of meaningful and semi-actual items, which reflects the specifications or environment of the population under test.
- Offer a unified description of meaningful data items to eliminate duplication or conflicting description of data items.
- Develop a set of efficient and powerful generation techniques based mainly on several statistical

methods. These techniques offer good flexibility to the user to generate different types and structures of meaningful data.
- Offer a set of test normality techniques to insure the efficiency of the generated data.
- Use a suitable intelligent soft computing such as fuzzy logic, neural network, or genetic algorithm in selecting the suitable data item structure and data generation technique.
- Increase the user's confidence in test data, testing, and application under test by allowing more participation to the user than recent test data generation tools; through the selection of the required list and its specifications, resource data and generation technique.
- Improve software industry by developing reliable software products.

3. TEST DATA GENERATOR MODULES

The intelligent generator for semi-actual test data consists of many modules, which include:

3.1 Setup Specifications

The setup specification is a very important preparation step before generation. The main activities of this step are shown in Fig. 1, which can be summarized as:
- Specifying the required list to be generated from the MDI sub-library.
- Specifying, or selecting, the suitable list structure from MDI structures sub- library using one of the intelligent selection techniques.
- Determining the list and fields specifications or selecting it from default specifications sub-library.
- Specifying the resource data or selecting it from default values sub-library.
- Specifying, or selecting, the suitable generation technique(s) using one of the intelligent selection techniques.
- Specifying the output file and device.

This module reflects the interface between the user and the meaningful data generation model to setup the required list and its related specifications and restrictions. It offers good flexibility and participation to the different users to select or insert the requirements for generating list(s) of meaningful data.

3.2 Data Descriptions

This module is used to offer a unified description for the meaningful data items, which are used by a set of applications that have the same or similar goal. The main steps for meaningful data description and library construction can be summarized as:
- Preparing a unified list by collecting all the data items and its related structures, from set of applications that have the same or similar goal, in a unified list(s).
- Sorting the unified list by sorting the contents of the unified list of data items.
- Eliminating duplicated items by selecting one data item from each set of duplicated items and deleting the others then links all structures for the deleted items into the selecting one.
- Eliminating similar items by selecting one item from each set of similar meaning items and deleting the others then linking all structures from the deleted items into the selected one. The produced data item called a pure list.
- Creating a MD library by storing all data items in the pure list in a MD items sub-library and the structures in MD item structures sub-library.

The result of this step is a meaningful data library, which contains:
- Meaningful data items (MDI) Sub-library: contains a set of possible meaningful data items that could be generated by this model.
- MDI Structures Sub-library: contains all possible structures for each element in the previous MDI sub-library.

The user has the ability to modify the library contents according to the application goal and environment. The main advantage of this step is to eliminate the duplicated or conflicted description(s) for meaningful data items used by the similar applications. It is a step towards standardizing the data description used in such applications.

3.3 Default Specifications and Values

This module represents the second part of the meaningful data library. It consists of two optional components that includes a:
- MDI default-specifications Sub-library (optional): contains the default specifications of each element in the previous MDI sub-library.
- MDI default samples Sub-library (optional): contains the default values or samples for each simple type of the elements in the MDI sub-library.

The importance of the defaults is to help the non-professional users in selecting the specifications and the values for the required data item.

3.4 Sample of Resource Data

The data generation process requires a set of resource data, which could be a sample of real data taken from the actual environment, set(s) of assumed data prepared by the expert or professional people who is working in the environment, default pre-saved data, pre-generated data, or sets of alphabets or boundary values data. Resource data is an important factor in this model and affects the efficiency of the generated data. The model focuses on using a sample of real or assumed data as a main resource. This generates data reflecting the population's specifications or cultures. The other resources of data are used as supporting resources. The resource data should validate according to data specifications used before by the generation engine. The main advantages of using a sample of real or assumed data as resource data can be concluded as increase:

- The ratio of meaningful (semi-actual) data, which reflects the specification, environment and culture of the population under test.
- The user participation in software development and testing. This could support the current trend of giving more participation the user and to eliminate the developer or database administrator bias in selecting the test data.
- The user confidence in test data, testing process and testing results.

3.5 Generation Techniques

This module uses a set of efficient generation techniques based mainly on several statistical methods. It offers the ability to generate lists of meaningful data, which are of different structures and volumes. The techniques include permutation with replacement, permutation without replacement, permutation with partial replacement, and permutation from multiple groups. These different permutation techniques produces (nk), (N!/(N-K)!), ((N-1)K + (n-1)K+1), and (N1*N2*N3* ….) permutations respectively [9, 10]. Besides the permutation techniques there are several statistical distributions, which include discrete binomial or multinomial distributions and continuous normal or Gamma distributions, and sampling random, systematic, or sequential techniques [11, 12]. The generation procedures are supported by hashing, sorting and searching techniques [13, 14]. This set of generation techniques offers flexibility to the users to generate the suitable volume of data. The generated data could be numeric, character or Boolean and representing simple, compound or composite structures. The selection of the suitable generation technique(s) is based on intelligent selection techniques. The generation techniques built as functions or routines are stored in a special library and ready when called by the generation engine.

3.6 Generation Engine

The generation engine represents the main processing part in the MD generation model. It generates the required list(s) of data based on the user setup to the list specifications and restrictions, which has been mentioned in setup step. This module uses information from MD library, resource data, generation techniques, and output requirements to generate the required data. There are two main strategies used for data generation. The first one is called one phase strategy, which is suitable to generate all types of lists. It requires that all fields in the selected structure be of simple type so it uses sampling or permutation techniques directly to generate the data in one phase. The second strategy is the multiple-phases, which is suitable to generate composite lists by generating its components in many phases. The result of each phase will be used as resource data for the next phases. The generation will continue until the required list is produced. The main steps for the meaningful test data generation phases are shown in Fig. 2, which can be summarized as:

i. Preparation phase:

- Select the required list to be generated.
- Select the suitable structure for the selected list, using one of the intelligent selection approaches.
- Setup list specifications.
- Setup field specifications for each field in the structure.
- Specify or insert the sample data for the field(s).
- Validate the sample data according to the field specifications.
- Store all specifications and sample data in temporary files to be used by the next phase.

ii. Generation phase:

- Read the preparation files.
- Call the main generation technique, according to one of intelligent selection approaches, to generate list of raw data.
- Filtering the raw list according to list specification.
- Testing the generated data according to the normality testing techniques.
- Customizing the generated list according to the customizing techniques to get the exact volume of data.
- Store the generated list on the specified output file within the specified output device.

3.7 Validation and Statistical Test

This module is responsible for validating the generated set(s) of data according to the list and field specifications. It is a descriptive testing of the generated data. The second type is statistical testing such as the normality testing and correlation coefficient. A common statistical technique judges whether an assumed model provides us with an adequate description of the observed data, it is a statistical test of the distributional assumptions built into the model. The power of any statistical test depends very much on the amount of information, which is available. The influence on the power is the detailed use of the data items, that is the nature of the test in terms of the criterion and the critical region. The tests for evaluating the assumed normality are [15]:

- The W- Test for normality
- An approximate analysis of variance test for normality.
- The probability plot correlation coefficient test for normality.
- The D-test for normality.

The multiple correlation coefficient (R) will reflect the measure of the linear association between the dependent variable Y and the independent variables x_1, x_2, .., x_k.

3.8 Output

This module is used to store the generated data in specified form, on the required output file and device according to specifications inserted by the user. The output device could be one of the I/O devices such as hard disk, floppy disk, CD, screen or printer. The importance of this module is to store the result in the required file and device for later usage.

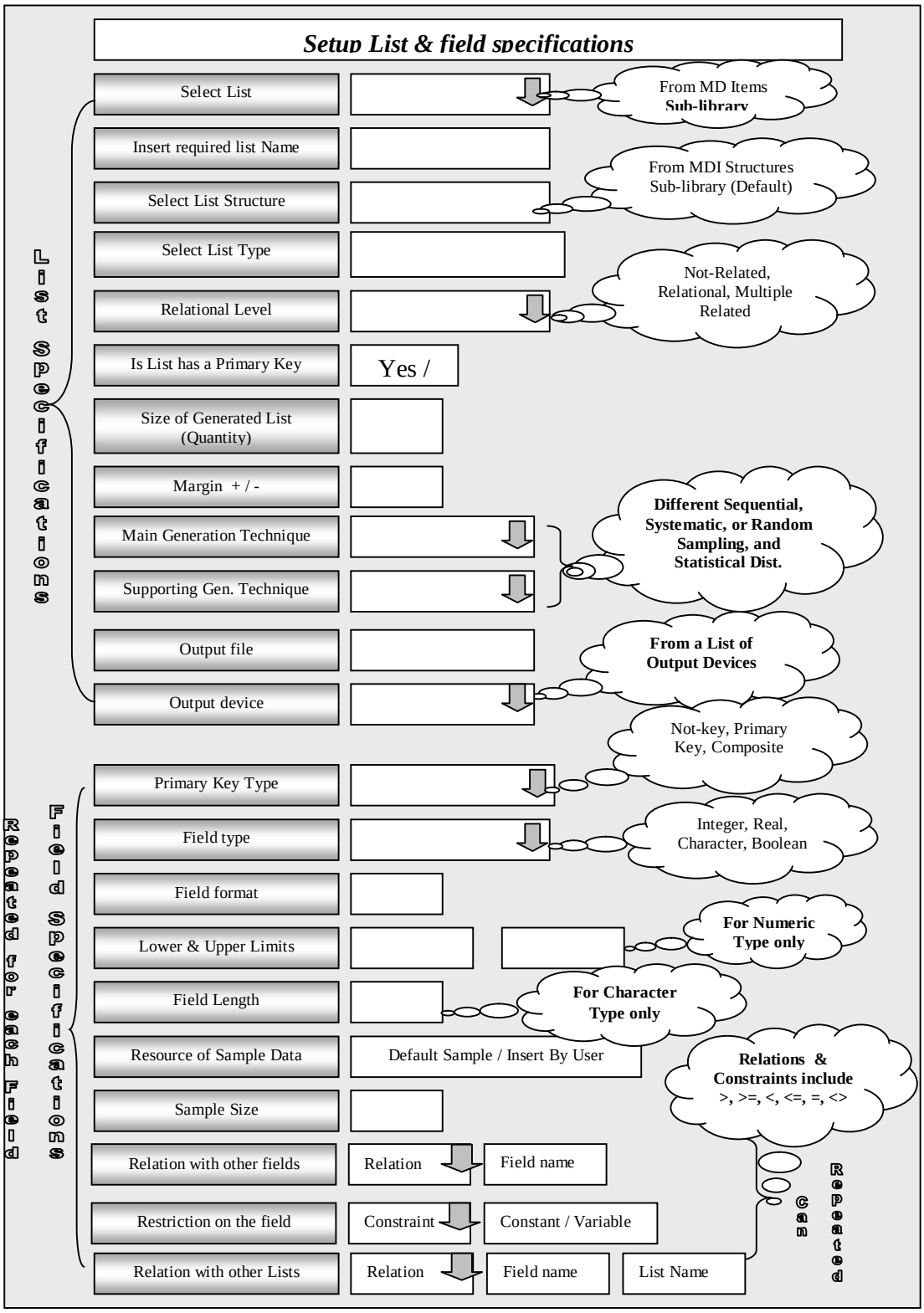

Fig. 1: Setup specifications for Data Item

.

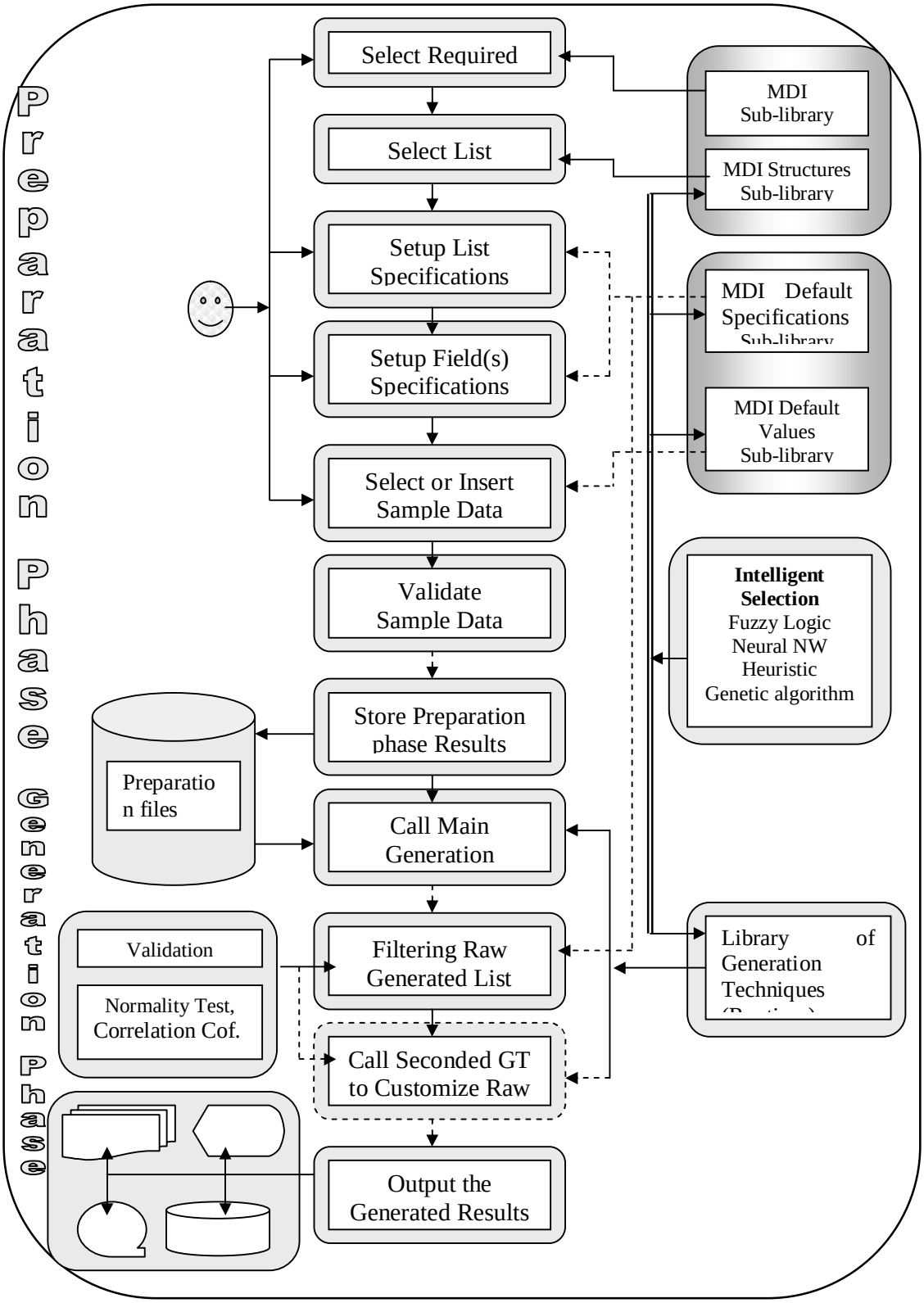

Fig. 2: Flow of semi-actual test data generator

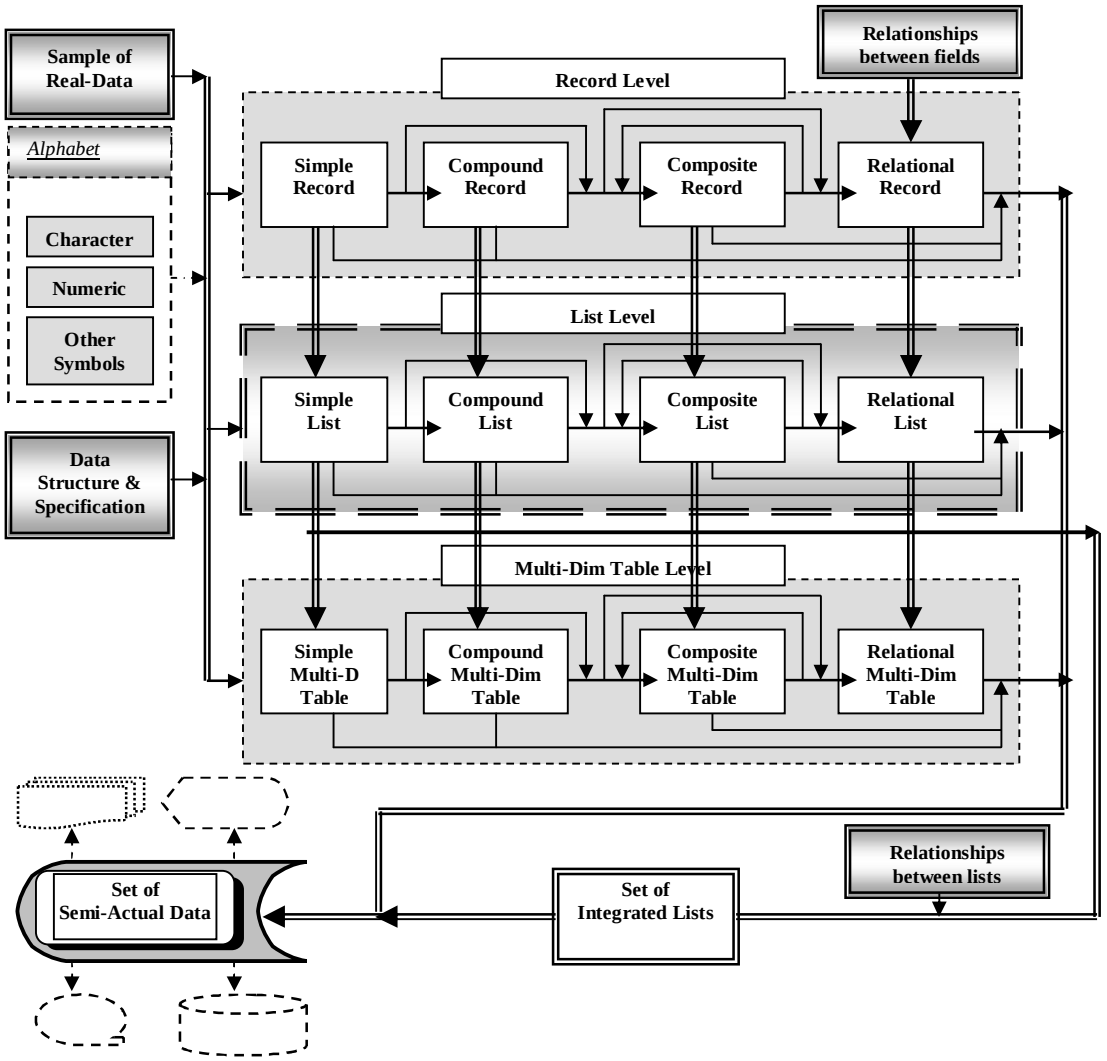

Fig. 3: Generation of Different Levels of Test Data

4. CONSTRUCTING DIFFERENT LEVELS OF DATA

To identify meaningful data there is a need for describing the different levels that represent the meaningful data. These levels could be classified mainly into single list, multiple dimension list, and integrated lists.

4.1 . Single List Level

The single list is constructed from a set of records that have the same type and structure. The list will hold the type of its base record; therefore the lists are classified into a simple, similar compound, not-similar compound, composite and relational types. The following is a brief description of different structures of records:

- Simple record consists of one field of integer, real, character, or Boolean type.
- Similar-Compound record consists of duplicating the same simple field many times.
- Not similar-Compound record consists of many simple fields but of different types.
- Composite record consists of aggregation of many simple, compound or composite fields.
- Relational record is a compound or composite record with a relationship(s) between some of its fields.

The construction of different records is shown in Fig. 3.

4.2 Multiple-Dimension List Level

This level is constructed from a collection of a set of the same type and structure lists. The table holds the same type of its base list, therefore the tables are classified into a simple, similar compound, not-similar compound,

composite, and relational. The generation of a multi-dimension table is done by repeating the process of generating a list for specific times according to the required table.

4.3 Integrated or Multiple Related Lists Level

Some of the software applications, specially the relational database, contain related lists or tables because there is some relationship(s) between these lists. The generation of this type requires specifying the relationships between these lists in accordance to its usage in the generation process.

5. CONCLUSION

The proposed model that overcomes some of the current problems to improve the test data generation. It offers good flexibility of the users, specially experts or professionals, to insert or select the required list of data to be generated, its structure and specifications, its components and their specifications, the generation technique(s), and resource of sample data. The usage of more powerful generation techniques, which are based on statistical methods besides the usage of real or assumed sample of data will participate in generating test data that holds a high ratio of semi-actual and meaningful items. The generated data will reflect the specification or environment of population under test. The intelligent selection of data item's structure and the generation technique will increase the model efficiency in selection, and increase total performance. The construction of meaningful data library will increase the efficiency of the test data generation process. The validation, normality test, and correlation coefficient will increase the efficiency and reliability on the generated data. The above results lead to increase the user confidence in the generated test data, testing process, testing result, and consequently in application under test. The ability of generating different types and structures of data taking in consideration the relationships between data fields will make this model suitable for generating test data for different applications specially the database applications, which are used in testing control systems.

REFERENCES

[1] B. T. Mynatt, Software Engineering with Student Project Guidance. USA, Prentice-Hall International Editions, 1990.

[2] R. S. Pressman, Software Engineering: A Practitioner's Approach. (4th Ed.), Singapore, McGraw-Hill Book Company, 1997.

[3] R. C. Ferguson and B. Korel, "The Chaining Approach for Software Test Data Generation". ACM Transactions on Software Engineering and Methodology, 5(1), pp.63 – 86, 1996.

[4] A. J. Offutt, Zhenyi J. Zhenyi, J. Pan, The Dynamic Domain Reduction Approach to Test Data Generation. Software Practice and Experience, to appear in 1999.

[5] A. J. Offutt, "An Integrated Automatic Test Data Generation System. Journal of Systems Integration", 1(3), pp.391-409, 1991,.

[6] S. A. Hameed, A. Deraman and A. Hamdan, A Framework for Database Test Data Generator, Technical Report FTSM / MEI LT- 48, University Kebangsaan Malaysia, 1998.

[7] S. A. Hameed, Meaningful Test Data Generation based on Statistical Methods, Ph.D. Thesis, Faculty of Information Science and Technology, University Kebangsaan Malaysia, 2000.

[8] L. H. Tsoukalas and R.E. Uhrig, Fuzzy and Neural approaches in Engineering, USA, John Wiley & Sons, Inc, 1997.

[9] L. Devore Jay, probability and statistics for engineering and the sciences, US Brooks / Cole Publishing Company, 3rd Edition, 1991.

[10] W. Hamming Richard, The Art of Probability for Scientists and Engineers. USA, Addison-Wesley Publishing Company, 1991.

[11] A. Agresti and B. Finlay, Statistical Methods for the social Sciences. San Francisco: Dellen Publishing Company, 1986.

[12] L. Lapin Lawrence, Statistics Meaning and Methods, Harcourt brace Jovanovich Inc, 1975

[13] M. A. Weiss, Data Structures and Algorithm Analysis in C, 2nd ed., Addison-Wesley Logman, Inc, 1997.

[14] C. A. Shaffer, A Practical Introduction to Data Structures and Algorithm Analysis. International Edition, Prentice-Hall International, Inc, 1997.

[15] A. A. Aziz, Simulation Systems for Statistical Tests, Ph.D. Thesis, University of Essex, UK, 1987.

BIOGRAPHIES

Shihab A. Hameed was Asst. Prof. at Electrical & Computer Eng. Department, Faculty of Engineering, IIUM University (currently in University of Bahrain). He obtained his Ph.D. from UKM university (Malaysia) in software engineering / SW testing. He has over twenty year industrial and educational experience in software development and as academician. He has published many research papers both locally and internationally.

Abdulmajid A. Al-Abbasi was Assoc. Prof. at Science in Engineering Department, Faculty of Engineering - IIUM University. Currently he is working at University of Bahrain. He obtained his Ph.D. from Essex university (UK) in Simulator Systems, 1988. His research interests are random number generation and analysis. He has several research papers published internationally.