

AETGSM Web: A Web Based Service for Automatic Efficient Test Generation from Functional Requirements

Siddhartha R. Dalal, Ashish Jain, Gardner Patton, Manish Rathi, and Paul Seymour
Bellcore
445 South Street
Morristown, NJ 07960
aetgweb@bellcore.com

Abstract

AETGSM Web is a web based service developed by Bellcore researchers for enabling model based testing. In model based testing, the functional test requirements of the system are first modeled and then testcases based on this model are created or generated. AETGSM Web employs a web based user interface to model the functional requirements of the System Under Test, and automatically generates testcases for the system using the AETGSM test case generation technology. A brief review of modeling using AETGSM Web and the AETGSM test generation paradigm is explained.

1 Introduction

As software systems become more complex and time to market becomes short, the deficiencies in ad-hoc and manual testing approaches become quite evident. One clear trend in testing in the future will be Model Based Automated Testing. In model based automated testing, the functional test requirements of the system are modeled first and then testcases based on this model are automatically generated. The AETGSM service employs unique technologies for modeling and test data generation. This paper presents highlights of these key technologies.

2 The AETGSM Test Generation Paradigm

The central idea* behind the AETGSM paradigm is the application of experimental designs to test generation. Each separate parameter of a test input tuple is considered a factor, with the different values for each parameter treated as

AETG is a trademark and service mark of Bellcore.

*Details on the AETGSM paradigm are beyond the scope of this paper. Interested readers are referred to [2, 3] for details.

level. The AETG generation algorithms ensure that each level of each factor is tested at least once with every other level of every other factor, which is called pairwise coverage of the input domain[†].

As an example, consider the functional testing of a input screen with 10 input fields. Even if one were to test each field with only one high and one low value, we would require 1024 test cases for exhaustively testing the screen. The AETGSM paradigm can efficiently test the screen using just 6 testcases which are guaranteed to cover all the pairwise interaction of the input domain. The reduction becomes more dramatic as the number of input parameters or the number of test values for these parameters increases.

3 AETGSM Web Service User Interface

Users interact with the service using a web browser. New users of the system use the web GUI widgets to input the functional requirements of the system. Advanced users can use a text-based notation to input the functional requirements. When viewed abstractly, the functional of a System Under Test consists of several input parameters, values for these parameters, and the constraints between the value combinations for these parameters.

Once the functional test requirements are input, AETGSM Web produces two sets of testcases: a valid set which contains the pairwise coverage of input domain and an invalid set which contains testcases which violate the requirement constraints. If the user provides illegal values for some or all the parameters then a third set of testcases covering the illegal input values is also produced. Invalid combinations and illegal value testcases can be used to test the robustness of the error handling functionality of the System Under Test.

[†]For sake of brevity, only pairwise generation is discussed here. The AETGSM test generation system can also generate efficient testcases for n-way interactions for $n > 2$.

```

field OrderDate InstallDate;
field AccountType LineType;

OrderScreen rel{

OrderDate: 19991231 20000101 20000103;
InstallDate: 19991231 20000101 20000103;
AccountType: Business Resident ;
LineType: PBX ISDN POTS T1 T3;

# Constraints
OrderDate < InstallDate;

If AccountType = Resident
then LineType != PBX T1 T3;
}

```

Figure 1. Text based model of System Under Test

OrderDat	InstallID	AccountT	LineType
19991231	20000101	Business	T3
20000101	20000103	Business	PBX
19991231	20000103	Business	T1
20000101	20000103	Business	T1
19991231	20000101	Business	T1
19991231	20000101	Business	ISDN
19991231	20000101	Resident	POTS
20000101	20000103	Business	T3
20000101	20000103	Business	POTS
20000101	20000103	Resident	ISDN
19991231	20000101	Business	PBX

Figure 2. Valid testcases generated by the AETGSM Web system

The text-based notation is best explained by the hypothetical example given in Figure 1.

In the hypothetical example, we are testing a “Order Screen” which has four fields OrderDate, InstallDate, AccountType, and LineType. The user is interested in generating pairwise-complete testcases using the test dates: 19991231, 20000101, and 20000103 for the first two fields; using the test values Business and Resident for the third field; and the test values ISDN, POTS, T1, and T3 for the fourth field. The first requirement constraint is the OrderDate is less than the InstallDate. The other requirement constraint states that a valid testcase in which the AccountType is Resident cannot have LineType as PBX, T1, or T3. Though not used in this example, the language supports boolean operators AND and OR as well.

The valid and invalid testcases generated by the system are shown in Figures 2 and 3 respectively. Besides generating testdata in the matrix form, AETGSM Web also produces JAVA class definitions to encapsulate the output testdata. To support easy use of these classes, AETGSM Web

OrderDat	InstallID	AccountT	LineType
19991231	20000101	Resident	PBX
20000101	20000103	Resident	T1
20000101	20000103	Resident	T3
19991231	19991231	Resident	ISDN
20000101	19991231	Resident	ISDN
20000101	20000101	Business	T3
20000103	20000103	Business	PBX

Figure 3. Invalid testcases generated by the AETGSM Web service

creates a template for a data driven main loop and a template for a procedure highlighting all the methods available to the user for accessing the individual testdata. This capability of the tool can be used by users who want to automate execution of their testcases using commercially available JAVA based test execution tools.

4 AETGSM Web Service Experience Reports

The AETGSM Web service is being used by many industrial users. The domains of the applications modeled include telecommunications, financial, system software, and email gateways. At Bellcore, we are using the service for testing numerous Operation Support and Network Systems. The AETGSM Web generated testcases have been very efficient in revealing more failures than testcases created manually. Reports of some actual experiences in using this technology in testing very large systems are available [1, 4, 3].

5 AETGSM Web Service Evaluation

More details on the service can be obtained at URL <https://aetgweb.tipandring.com/AboutAETGweb.html>. Free trial access to the system can be obtained by emailing a request to aetgweb@bellcore.com.

References

- [1] K. Burr. Combinatorial test techniques: Table-based, test generation, and code coverage. In *Software Testing Analysis and Review (STAR West)*, San Diego, CA, Oct. 1998.
- [2] Cohen, Dalal, Fredman, and Patton. The AETG system: An approach to testing based on combinatorial design. *IEEE Transactions on Software Engineering*, 23(7), 1997.
- [3] Cohen, Dalal, Parelius, and Patton. The combinatorial design approach to automatic test generation. In *Proceedings of the Seventh International Symposium on Software Reliability Engineering*, White Plains, NY, 1996.
- [4] Dalal, Jain, Karunanithi, Leaton, and Lott. Model-based testing of a highly programmable system. In *Proceedings of the Ninth International Symposium on Software Reliability Engineering*, Paderborn, Germany, Nov. 1998.