

Serial No. 713,896
Filing Date 13 September 1996
Inventor Timothy M. Bearse
Edward L. Gagnon
Michael L. Lynch
James Souza

NOTICE

The above identified patent application is available for licensing. Requests for information should be addressed to:

OFFICE OF NAVAL RESEARCH
DEPARTMENT OF THE NAVY
CODE OCCC3
ARLINGTON VA 22217-5660

DTIC QUALITY INSPECTED 2

19970115 080

1 N.C. 77513

2
3 A MODEL-BASED PROCESS FOR TRANSLATING TEST PROGRAMS

4
5 STATEMENT OF GOVERNMENT INTEREST

6 The invention described herein may be manufactured and used
7 by or for the Government of the United States of America for
8 governmental purposes without the payment of royalties thereon or
9 therefor.

10
11 BACKGROUND OF THE INVENTION

12 (1) Field of the Invention

13 The invention relates to computerized test programs and is
14 directed more particularly to translation of a test program from
15 a first computer language to a test program in a second computer
16 language.

17 (2) Description of the Prior Art

18 A test program consists of a well-defined sequence of
19 instructions in a specific test-based computer language which
20 results in an automatic test station (1) applying stimuli to a
21 unit under test, and (2) recording measurements of various
22 parameters received responsive to the stimuli to confirm that the
23 parameters are within specified tolerances. The test program
24 further includes branching instructions which are used to control
25 the flow of the program.

1 Test programs are currently translated from one computer
2 language into another by converting test program code, line by
3 line, from the existing test language into the language of the
4 target automated test system. There are substantial
5 disadvantages to this approach: (1) typically, line by line
6 translation fails to take into account the test strategy employed
7 when executing the existing test program; (2) line by line
8 translations are inadequate when changes are needed to the
9 translated test program; (3) it is very difficult to assess the
10 performance and validity of the translated test program; and (4)
11 current translation processes, whether manual or automated,
12 generally are labor intensive, and/or too costly, and/or simply
13 ineffective.

14 Accordingly, there is a need for a novel process for
15 translating test programs from one computer language into a
16 second computer language, which process includes retention of the
17 test strategy employed in the existing test program, is not
18 totally dependent upon a line by line translation, which results
19 in a new language test program which is relatively easy to assess
20 from a performance and validity standpoint, and which is
21 relatively less labor intensive and less costly than processes
22 employed heretofore.

23 24 SUMMARY OF THE INVENTION

25 It is, therefore, an object of the invention to provide a
26 novel model-based process for translating test programs from one

1 computer language into a second computer language, which process
2 retains the test strategy employed in the existing test program,
3 is only partially dependent upon a limited line by line
4 translation, and which results in a new language test program
5 which is relatively easy to assess from performance and validity
6 standpoints, and which is relatively less labor intensive and
7 less costly than the heretofore employed processes.

8 With the above and other objects in view, as will
9 hereinafter appear, a feature of the present invention is the
10 provision of a model-based process for translating test programs
11 from a first computer language to a second computer language, the
12 process comprising the steps of extracting test strategy and
13 replaceable item callouts from an existing test program in the
14 first language, converting the extracted test strategy and
15 replaceable item callouts from the existing test program into an
16 asymmetric dependency model, converting the asymmetric dependency
17 model into a model-based test strategy, extracting code segments
18 from the existing test program for individual translation,
19 translating the extracted code segments into the second language,
20 and merging the model-based test strategy and the translated code
21 segments into a new test program.

22 The above and other features of the invention, including
23 various novel details and combinations of process steps, will now
24 be more particularly described with reference to the accompanying
25 drawings and pointed out in the claims. It will be understood
26 that the particular process embodying the invention is shown and

1 described by way of illustration only and not as a limitation of
2 the invention. The principles and features of this invention may
3 be employed in various and numerous embodiments without departing
4 from the scope of the invention.

5 6 BRIEF DESCRIPTION OF THE DRAWING

7 Reference is made to the accompanying drawings in which is
8 shown an illustrative embodiment of the invention, from which its
9 novel features and advantages will be apparent.

10 In the drawings:

11 FIG. 1 is a flow chart depicting one form of process
12 illustrative of an embodiment of the invention; and

13 FIG. 2 is a flow chart illustrative of a test strategy and
14 replaceable item callouts.

15 16 DESCRIPTION OF THE PREFERRED EMBODIMENT

17 The purpose of this invention is to provide a process for
18 translating test programs from one computer test language into a
19 completely different computer test language. The invention
20 provides a process employing model-based technology for
21 translating existing test programs between two different test
22 languages such that the translated test program may be executable
23 on re-targeted automatic test stations.

24 Referring to FIG. 1, it will be seen that one step 10 of the
25 inventive process 12 involves extracting test strategy and
26 replaceable item callouts from an existing test program 14. In

1 this first step 10 of the process 12, the existing test program
2 14 is parsed to extract the underlying test strategy and
3 replaceable item callouts at each terminal node in the test
4 strategy.

5 By "parsing" is meant extracting the test strategy, which
6 entails (1) identifying the code segments for performing the
7 tests, and (2) identifying the branching, or flow, between the
8 code segments. The test strategy may be extracted by any
9 selected one of several known techniques. The test strategy is
10 the structure, i.e., the sequence and flow, of the tests, that
11 is, the "test tree" or underlying "if - then" (i.e., PASS/FAIL)
12 structure inherent in the test program. The test strategy
13 contains the sequence of tests to be executed, as well as the
14 appropriate branching decisions, based on the outcome of each
15 test. Replaceable item callouts are used to identify the base
16 level modules/components of a device or unit under test that are
17 identified for replacement or repair by the tests within a given
18 test strategy.

19 In FIG. 2, there are shown a test strategy for testing a
20 personal computer system, and replaceable item callouts. The
21 test strategy of FIG. 2 includes three tests and identifies, at
22 the terminal nodes of the test strategy, three components of the
23 system for repair or replacement. These three components, the
24 keyboard, monitor, and disk drive, identified for repair or
25 replacement at the terminal nodes, are the replaceable item
26 callouts.

1 The extracted test strategy and replaceable item callouts
2 are used to generate an asymmetric dependency model. In this
3 step 16 of the process 12, the extracted test strategy and
4 replaceable item callouts from the existing test program 14 are
5 converted into an asymmetric dependency model. A dependency
6 model of a unit under test represents the inferences or
7 conclusions that can be drawn when a test passes or when a test
8 fails. In this type of model, each test has two diagnostic
9 inference lists associated with it: (1) a list of failure modes
10 that can be cleared when the test passes, and (2) a list of
11 failure modes that are to be suspect when the test fails. An
12 asymmetric dependency model results when the two diagnostic
13 inference lists associated with any specific test do not contain
14 the same failure modes.

15 In extracting the test strategy and replaceable item
16 callouts from an existing test program to generate the asymmetric
17 dependency model, a one-to-one relationship exists between the
18 replaceable item callouts and the failure modes. The two
19 diagnostic inference lists are created for each specific test
20 within the test strategy by analyzing the test strategy and
21 replaceable item callouts. The list of failure modes that can be
22 cleared when a specific test passes can be compiled by
23 identifying all the failure modes that will cause the test to
24 fail and building a list of these identified failure modes.

25 The list of failure modes that are to be suspect when a
26 specific test fails is built by assembling a list of all possible

1 failure modes for the test strategy and then removing from the
2 list of possible failure modes any failure mode which will not
3 cause the specific test to fail.

4 Referring to FIG. 2, it will be seen that in the test
5 strategy shown, a list of failure modes that can be cleared when
6 TEST 3 passes comprises the disk drive. The list of failure
7 modes that are to be suspect when TEST 3 fails comprises the
8 keyboard and the disk drive. For TEST 2, the list that can be
9 cleared when passed comprises the keyboard, and the list of
10 failure modes that are suspect when TEST 2 fails comprises the
11 keyboard monitor and disk drive. For TEST 1, the list that can
12 be cleared if TEST 1 passes is the monitor and disk drive, and
13 the list of failure modes that are suspect when TEST 1 fails are
14 the monitor and disk drive. Thus, the dependency lists for TEST
15 1 are not asymmetric.

16 After generating the asymmetric dependency model, the next
17 step 18 is to convert the asymmetric dependency model into a
18 model-based test strategy. The asymmetric dependency model is
19 analyzed and a model-based test strategy generated.

20 Simultaneously with the extraction of test strategies and
21 replaceable item callouts from the existing test program in step
22 10, step 20 may be undertaken, which involves extracting code
23 segments from the existing test program. In this step 20 of the
24 process 12, code segments from the existing test program 14 are
25 identified and extracted for individual translation.

1 Following the extraction 20 of selected code segments from
2 the existing test program 14, each individual code segment is
3 translated 22 into the new language using either manual or
4 automated techniques.

5 After translation 22 of code segments from the existing test
6 program 14 and after generation 18 of a test strategy, the
7 translated code segments, which are in the new test language, are
8 merged 24 with the model-based test strategy to form 26 a new
9 test program which may then be compiled and executed 28 on the
10 target automatic test station (not shown). The merger of the new
11 test strategy and the translated code segments is effected in
12 three steps, including (1) reading the new test strategy, (2)
13 creating the branching, or flow, structure, and (3) entering the
14 translated code segments into the new flow structure.

15 The proposed process uses the concepts of a fixed test
16 strategy and an asymmetric dependency model in order to translate
17 existing test programs into new test programs.

18 The advantages of this process become apparent when one
19 needs to make changes to the new test program after translating
20 to the new test language on a different test station. This new
21 test program translation process is model-based, which means that
22 when the translated test program is complete, one has a model
23 that is representative of the test program. The primary
24 advantages in this process are the ability to assess the
25 performance and validity of the model, which reflects the new
26 test program, using model-based testability analysis tools, and

1 the ability to easily change the model, and subsequently the test
2 program, in addition to the ability to assess these changes
3 before assembling the final executable test program.

4 The impact of adding to, or deleting tests from, the test
5 program can be assessed before it is integrated into a test
6 station. Cost savings are achieved using this process because
7 integration of the test program and an interface device with a
8 test station is not necessary until the parameters associated
9 with the test program are considered to be satisfactory. In
10 addition, the model-based nature of the process enables the test
11 program developer to optimize time and money expended in the
12 translation effort.

13 The means for implementing any of the steps in the process
14 can be automatic, using software, or manual.

15 It is to be understood that the present invention is by no
16 means limited to the particular series of steps herein disclosed
17 and/or shown in the drawings, but also comprises any
18 modifications or equivalents --

1 N.C. 77513

2
3 A MODEL-BASED PROCESS FOR TRANSLATING TEST PROGRAMS

4
5 ABSTRACT OF THE DISCLOSURE

6 A model-based process for translating test programs from a
7 first computer language to a second computer language includes
8 the steps of extracting test strategy and replaceable item
9 callouts from an existing test program in the first language,
10 converting the extracted test strategy into an asymmetric
11 dependency model, converting the dependency model into a model-
12 based test strategy, extracting code segments from the existing
13 test program, translating the extracted code segments into the
14 second language, and merging the model-based test strategy and
15 the translated code segments into a new test program in the
16 second language.

10

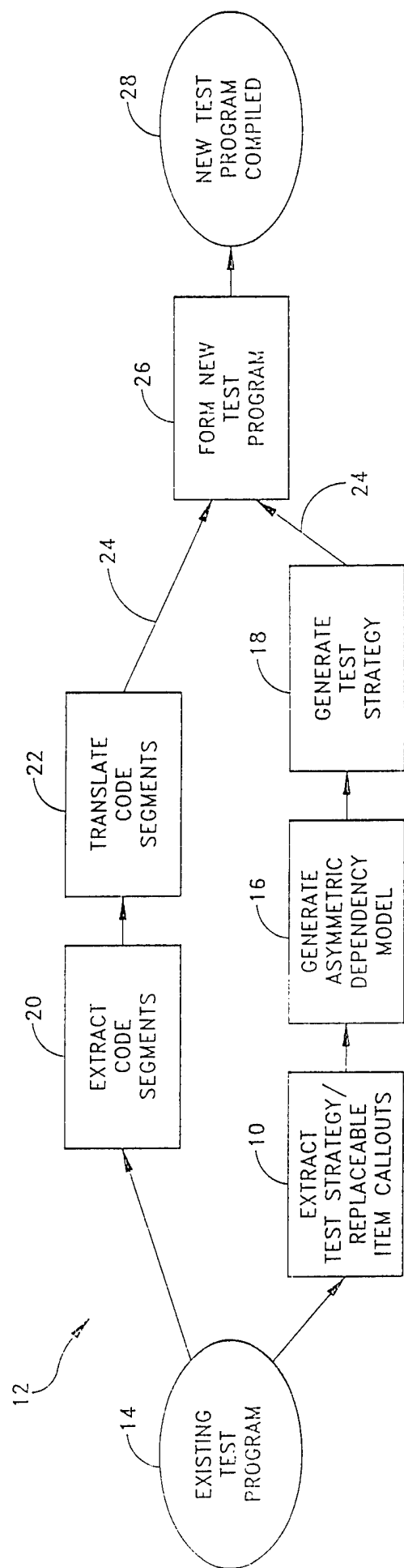


FIG. 1

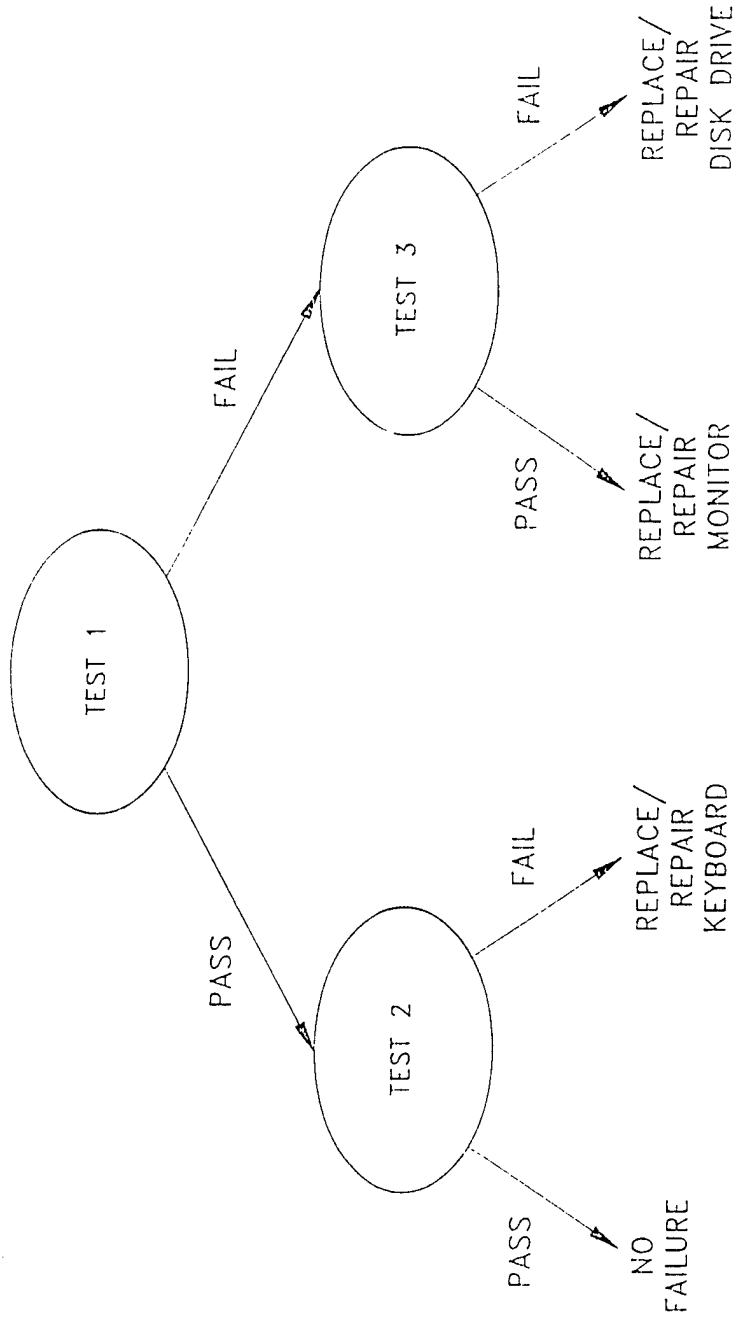


FIG. 2