XA9949629

RETRANS - A TOOL TO VERIFY THE FUNCTIONAL EQUIVALENCE OF AUTOMATICALLY GENERATED SOURCE CODE WITH ITS SPECIFICATION

H. MIEDL
Institut für Sicherheitstechnologie (ISTec) GmbH
Garching, Germany

Abstract

Following the competent technical standards (e. g. IEC 880) it is necessary to verify each step in the development process of safety critical software. This holds also for the verification of automatically generated source code. To avoid human errors during this verification step and to limit the cost effort a tool should be used which is developed independently from the development of the code generator. For this purpose ISTec has developed the tool RETRANS which demonstrates the functional equivalence of automatically generated source code with its underlying specification.

Introduction

Along with the technological change in instrumentation and control technique in all industries, analogous (hardwired) systems are presently more and more replaced by digital, mostly computer based systems. This concerns increasingly also I&C functions important to safety in nuclear power plants.

An important feature of the computer based instrumenation and control technique is the realisation of functionality by software. The software development process is often characterised by the automatic generation of source code from a formal specification which is stored in a database. Therefore, an important part within the software life cycle is taken over by a code generator (see fig. 1).

Within ISTec a method was developed to demonstrate the correct transformation of a formal specification into source code (i. e. the correct performance of the code generator). The basic concept is the analysis of the generated source code to reconstruct its inherent functionality and to compare it with its underlying specification to demonstrate functional

equivalence between both. During the analysis of the source code no transformation rules of a specific code generator are considered, rather the analysis is based on the structure of the generated source code only. The information gathered during analysis of the generated source code about its inherent functionality can be used in addition for plausibility controls with regard to redundancies. This supports the discovery of possible specification errors. Furthermore, the tool reveals inconsistencies, both in the source code and the database of the specification. In the database redundant information is checked for conformance. The source code is examined with regard to the conformance of the code with the syntactical requirements of its program structure.

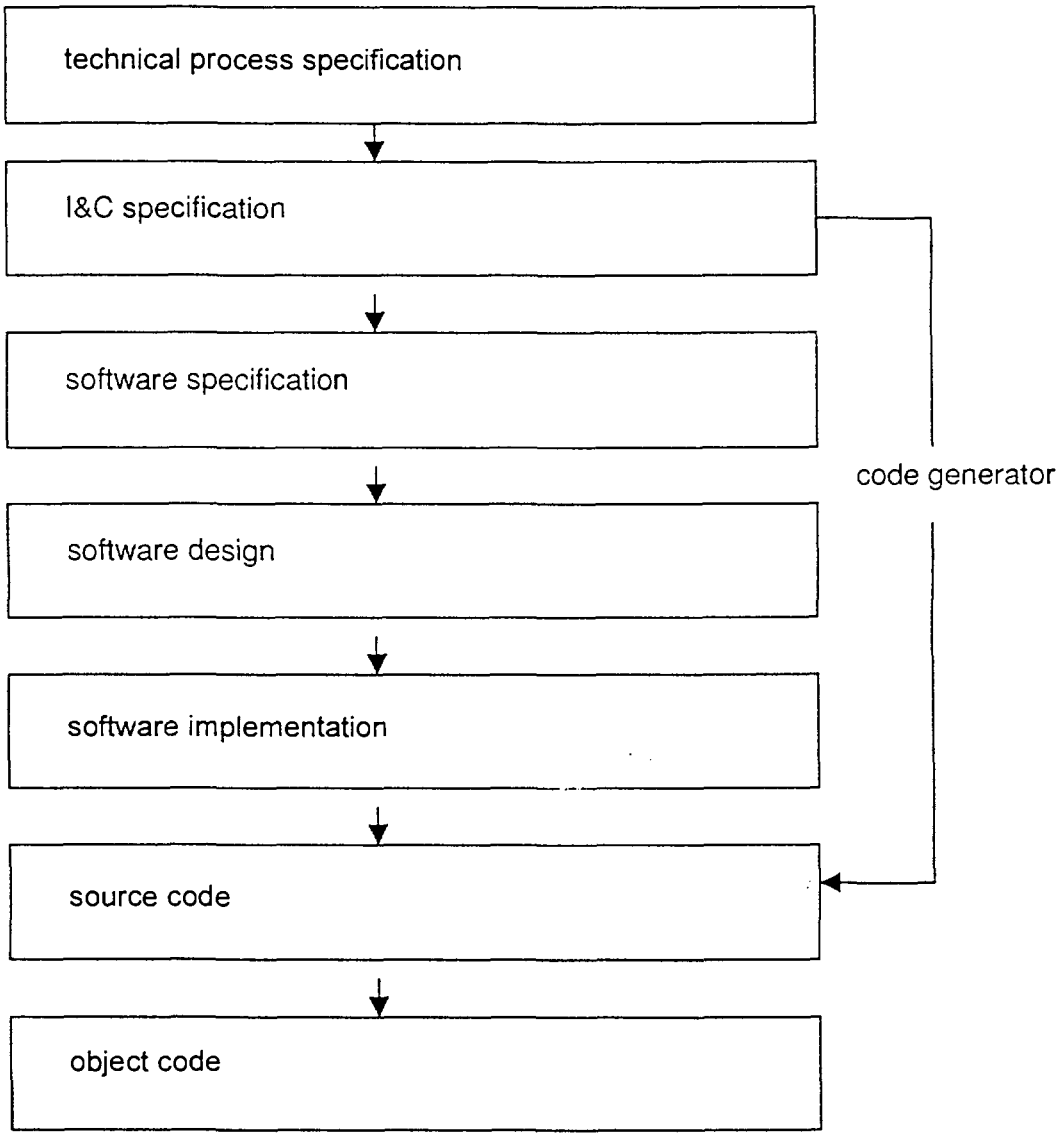

Fig. 1: Software development process (code generation skips several steps of a conventional software development process)

Digital I&C system TELEPERM XS

The digital I&C system TELEPERM XS of Siemens/KWU uses for the specification and implementation of the software for the processing of I&C functions on computers an inhouse developed tool, SPACE (specification and coding environment). With the assistance of a graphical interface (SPACE editor), I&C functions (functional diagrams FD) are constructed from prefabricated, normed basic elements (function blocks FB) and are organised as groups (functional diagram groups FDG) which are executed cyclically on single processing units

(see fig. 2). These graphical specifications are stored in the form of database tables (SPACE database), that contain - in the sense of a formal specification - already the complete information for the functionality of the computer programs. The associated code generation is based only on the database tables, and uses and includes only prefabricated FB modules and declaration files. Following predefined rules the code generator creates automatically the C programs (FD and FDG modules).

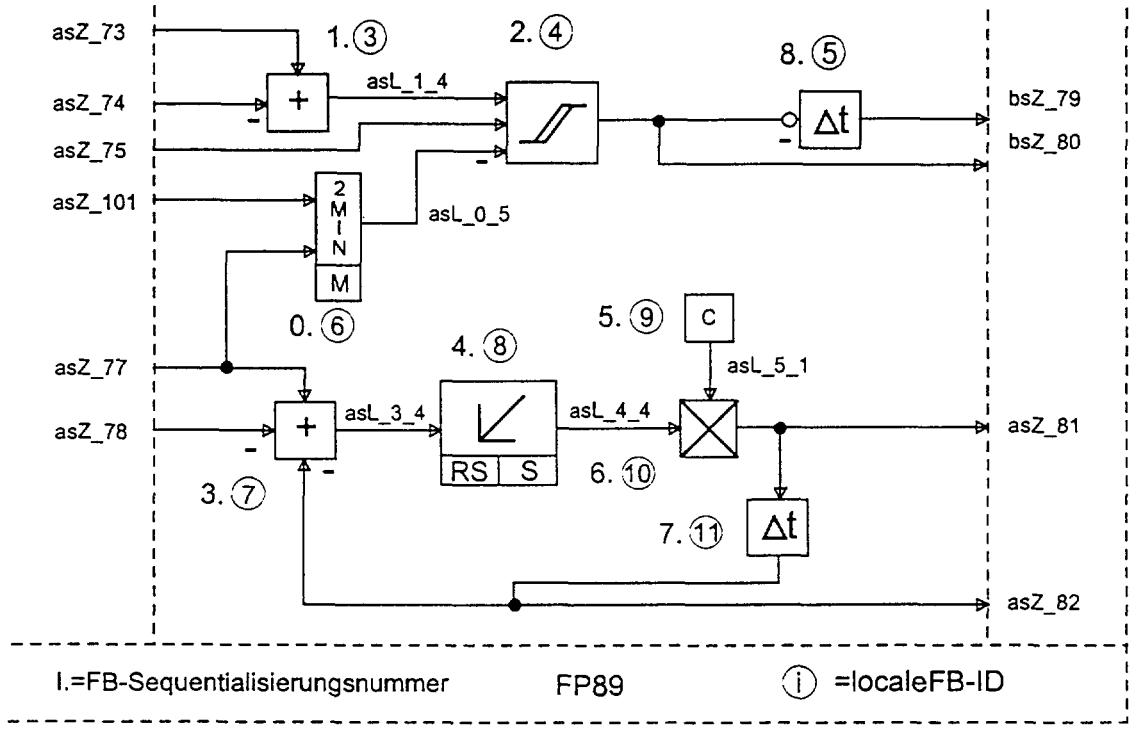

Fig. 2: Pseudo functional diagram

Therefore an important part of the application software development within the software life cycle is performed by a specific tool (FDG code generator). Besides the adequate specification of the FD and FDG (reproducing the requirements of the technical process) and the correct implementation of the basic elements (FB), the proper operation of the application programs (FD and FDG modules) depends additionally on the reliable operation of the FDG code generator. Therefore, qualifying the I&C application program as a primary aim has to be done by the verification of the transformation procedure of the code generator.

This task is taken over by the tool RETRANS.

Concept of RETRANS

The basic concept of the tool RETRANS is the reverse transformation of the functions contained the generated C source codes (FD and FDG modules) into a form which enables an automatic analysis of the equivalence with the original specification presented in the form of database tables. That means, RETRANS does not rely on the rules of the FDG code generator, rather it relies on the program structure of the FD and FDG modules and on the data model of the SPACE database (see fig. 3).

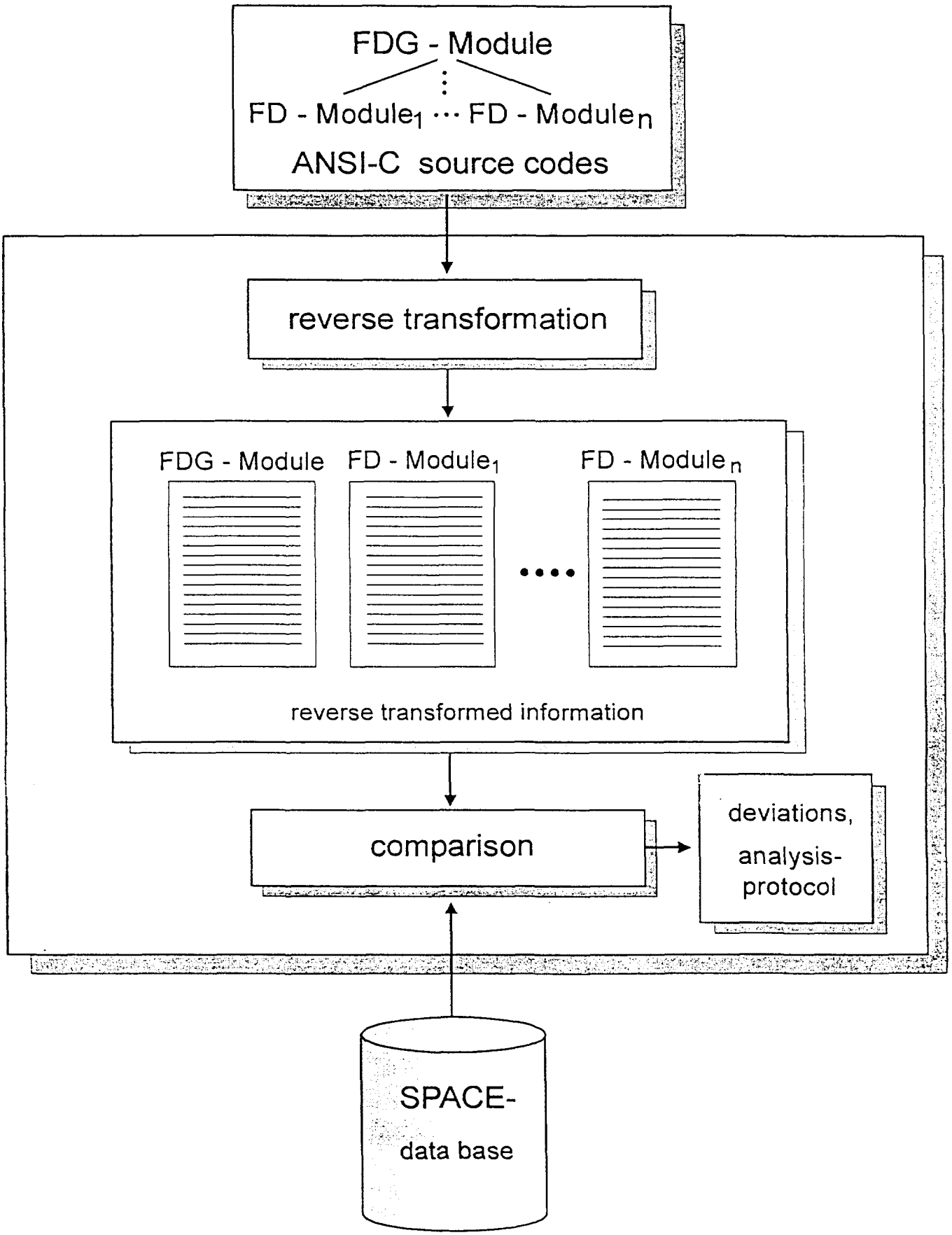

Fig. 3: Concept of the tool RETRANS

As a result of the reverse transformation step a file is created from every FDG module and FD module which contains a complete description of its function (see fig. 3). The form of these files is already processed for the following comparison with the database tables of the specification and is structured according to analysis items which are character strings of C source code. The comparison of the analysis items with the SPACE database is performed with respect to consistency and completeness. RETRANS generates analysis protocols, both on screen and in paper form.

The following analysis items are investigated:

- internal FDG identification string
- external FDG identification string
- input function of the FDG module
- route function of the FDG module
- output function of the FDG module
- period of the FDG
- compute function of the FDG module
- internal FD identification string
- external FD identification string
- not connected FB input ports of the FD modules
- not connected FB output ports of the FD modules
- unchangeable FB parameters of the FD modules
- changeable FB parameters of the FD modules
- compute function of the FD module

The functional diversity of the tool RETRANS from the FDG code generator, i. e the independence from its transformation rules for code generation, enables the tool to reveal errors in these rules which could not be detected by a tool using the inverted generation rules for the reverse transformation.

Furthermore RETRANS creates during the reverse transformation among other findings intermediate result files which represent in a compact form the FD and their functionality. These intermediate results can be used to apply tool based plausibility controls on redundant FD to detect potential specification errors as for example deviations in parameter values.

Summarising the essential performances of the software analysis tool RETRANS can be described as follows:

- automatic comparison of the specification of the application programs stored in a database with the functionality of the automatically generated C source code.
- hints for the analyser with respect to the plausibility of FB parameters in redundant channels.
- hints concerning inconsistencies in the database.
- hints concerning inconsistencies in the C source code.

Example

An example demonstrates the use of RETRANS. It is taken from the TELEPERM XS laboratory system of GRS/ISTec. The error detected by RETRANS was forced on purpose.

Fig. 4 shows a part of a functional diagram. It is a test function to switch a lamp on and off. After code generation it was decided to change the parameter `fl_1_1` to the value of 2 seconds. This change was done with the SPACE editor but without new generation of the software. Therefore, the old source code (see fig. 5) and the database entries are different in the parameter `fl_1_1`.

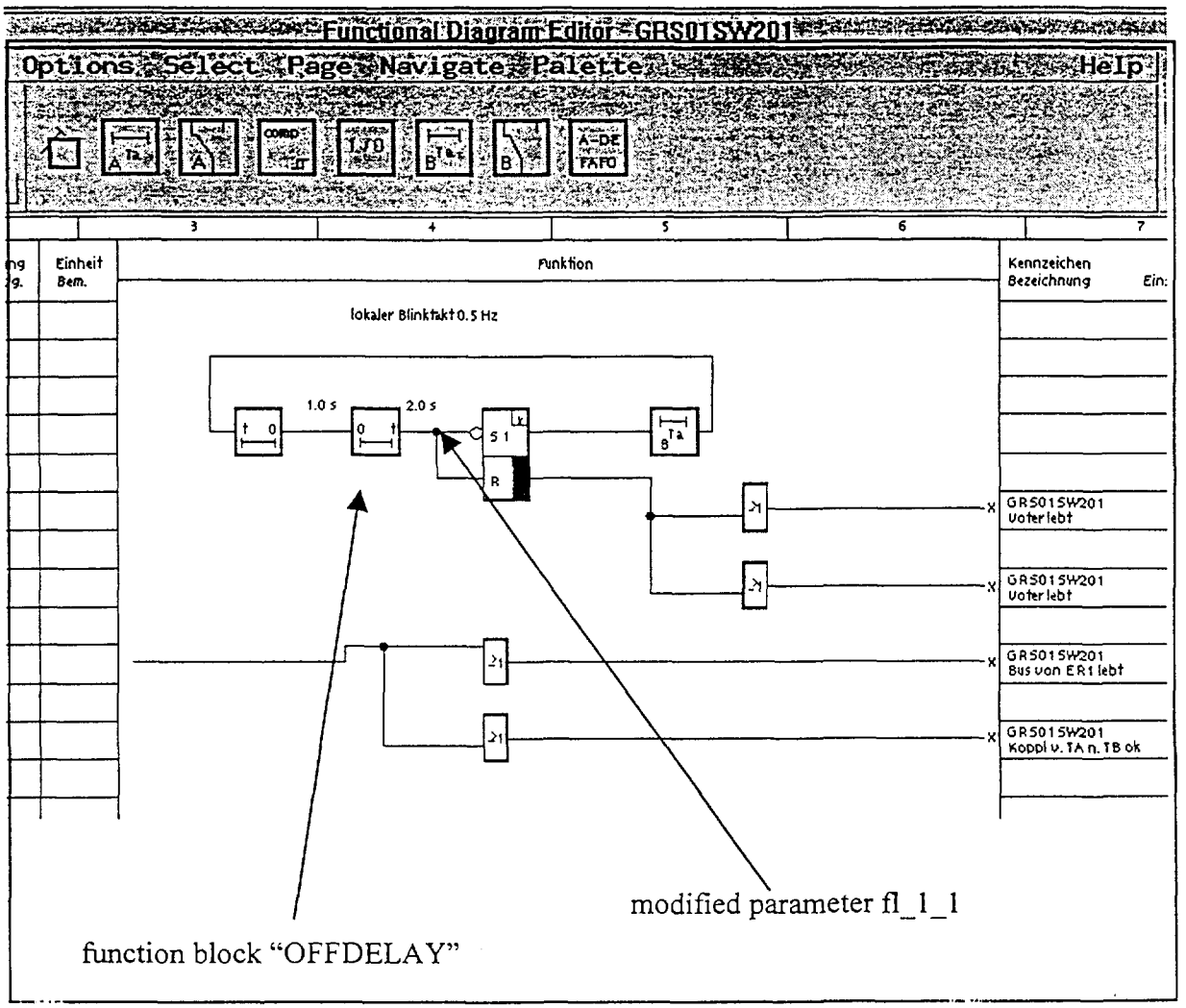

Figure 4: Example functional diagram with changed parameter

```
/* ------------------------------------------------------------------ */
.
/* file        : fd_3.c                                               */
/* function    : i_fd_3_Compute                                       */
.
/*.FD fd_id    : 3                                                    */
/* FD KKS      : GRS01SW201                                           */
/* FD version  : 01.00                                                */
/* FD changed  : 07.05.97 11:03:37                                    */
.
/* FDG generator version: 02.20                                       */
/* FDG generator changed: 09.01.97                                    */
/* File generation date : 12.05.97 16:59:21                           */
/* ------------------------------------------------------------------ */
.
.
.
     /* Initialisation of internal FD identification string */
static const fdIdString_t fd_3_IntIdent_p =
  /* "FD<fd_id>/<fd_version>/<fd_changded>/                  */
  /*            <fd_generator_version>/<fd_generator_changed>" */
  "FD00003/01.00/07.05.97/02.20/09.01.97";
.
.
.
     /* Initialisation of changeable FB parameters */
static const fd_3_CParams_t fd_3_CParamsConst = {
  /* <param_value>     ,    <type>_<fb_seq>_<param_nr> */
  1.0,                    /* fl_0_1 */
  1.0,                    /* fl_1_1 */
};
.
.
.
/* Function implementing the FD module                                */
.
void i_fd_3_Compute()
{
.
.
.
  static const fb_353_t locFb_1 = { /* Name "OFFDELAY", loc_id 6, Page 1 */
.
.
.
     &fd_3_CParams.fl_1_1,
.
.
.
  };
.
.
.
     /* Call FB modules: */
.
.
.
  r = g_fb_353   ( &locFb_1 );       /* OFFDELAY     , 1, 4A */
  if (r != OK) i_fdgAppendFbRetCode( 3, 1, r );
.
.
.
} /* i_fd_3_Compute() */
```

Figure 5: Source code of the functional diagram before changing the parameter (partly)

```
(    fdg_id):                                                    0014
(fdgversion):                                                    01.01
(fdgchanged):                                                 07.05.97
(cg_version):                                                    02.20
(cg_changed):                                                 09.01.97
(fdidstring):                          FD00003/01.00/07.05.97/02.20/09.01.97

 Funktionsplangruppe: 14
 Fehlercode 11 mit 07.05.97 und 14.10.97

(  fdginput):                                    3_CSignals.bsZ_4/bsInitError
(  fdginput):       3_CSignals.bsZ_4/t_101_bsZ_4.s/bsInitError/t_101_bsZ_4
(  fdginput):                                   3_CSignals.bsZ_16/bsInitError
(  fdginput):    3_CSignals.bsZ_16/t_101_bsZ_16.s/bsInitError/t_101_bsZ_16
.
.
.
( fdgoutput):                                d_20046->bs_2/3_CSignals.bsZ_11
( fdgoutput):                                d_20046->bs_3/3_CSignals.bsZ_6
( fdgoutput):                                d_20046->bs_4/3_CSignals.bsZ_19
( fdgoutput):                                d_20046->bs_5/3_CSignals.bsZ_20
(fltastring):                                                    flta/0.025
(   fdpart0):                                                     3_Compute


(   plan_id):                                                       00003
(fd_version):                                                       01.00
(fd_changed):                                                    07.05.97

 Funktionsplan: 3
 Fehlercode 1 mit 07.05.97 und 14.10.97

(cg_version):                                                       02.20
(cg_changed):                                                    09.01.97
(fbidstring):                                          302/01.00/30.03.93
(fbidstring):                                          351/01.00/16.08.93
(fbidstring):                                          353/01.00/18.01.94
(fbidstring):                                          354/01.00/18.01.94
(fbidstring):                                          356/01.00/18.01.94
(  usignals):                                                bs_302_1/0/0
(  usignals):                                                bs_302_2/0/0
(  ysignals):                                                     ms_2_5
(   cparams):                                                   fl_0_1/1.0
(   cparams):                                                   fl_1_1/1.0

 Funktionsplan: 3
 Fehlercode 50 mit 1.0 und 2.0

(   compute):                                                       fb_354
(   compute):                                          CSignalse.bsL_13_2
(   compute):                                           CSignalsa.bsL_0_2
(   compute):                                              CParams.fl_0_1
(   compute):                                              DParams.ui_0_1
(   compute):                                                     fb_302
.
.
.
(   assign4):                                                       302/11
(   assign4):                                                       302/12
(   assign4):                                                    351_w/13

Analyse erfolgreich beendet! Es wurden  3 Unstimmigkeiten entdeckt
```

Figure 6: RETRANS analysis protocol (partly)

The RETRANS analysis protocol (Fig. 6) shows three anomalies. The first one was found in the functional diagram group 14 and shows different dates of changing the functional diagram 3 (Fehlercode 11). The second one was found in the functional diagram 3 and shows also the different dates of changing the functional diagram 3 (Fehlercode 1). It is the same detected cause of the first anomaly. The third one (Fehlercode 50) shows the difference of the value of "fl_1_1" 1.0 in the program and 2.0 in the database (and that means in the graphical representation).

In this example the failure is corrected by a new code generation.

Another example shall demonstrate the operation of the plausibility control with regard to redundancies.

```
Vergleich von Plan 1 (KKS 1) und Plan 2 (KKS 2) beginnt
 FPG-Id zu Plan 1: 1 und Name zu Plan 1: HKMP 1
 FPG-Id zu Plan 2: 2 und Name zu Plan 2: HKMP 1

 (fbseq 0):     txArray_fbseq_1/ (fbseq 0):     txArray_fbseq_1/
 (fbseq 0):       fl_fbseq_2/0.0 (fbseq 0):       fl_fbseq_2/0.0
 (fbseq 0):        fl_fbseq_3/50 (fbseq 0):        fl_fbseq_3/50
 (fbseq 0):       fl_fbseq_4/0.0 (fbseq 0):       fl_fbseq_4/0.0
 (fbseq 0):       fl_fbseq_5/500 (fbseq 0):       fl_fbseq_5/500
 (fbseq 1): txArray_fbseq_1/FB5 (fbseq 1): txArray_fbseq_1/FB5
 (fbseq 1):       fl_fbseq_2/220 (fbseq 1):       fl_fbseq_2/220

 ********** WARNING **********  (fbseq 1):      fl_fbseq_3/25.5 no match

 Vergleich von Datei 1 und Datei 2 beendet

 Vergleich von Plan 2 (KKS 2) und Plan 1 (KKS 1) beginnt
 FPG-Id zu Plan 2: 2 und Name zu Plan 2: HKMP 1
 FPG-Id zu Plan 1: 1 und Name zu Plan 1: 4 HKMP 1

 (fbseq 0):     txArray_fbseq_1/ (fbseq 0):     txArray_fbseq_1/
 (fbseq 0):       fl_fbseq_2/0.0 (fbseq 0):       fl_fbseq_2/0.0
 (fbseq 0):        fl_fbseq_3/50 (fbseq 0):        fl_fbseq_3/50
 (fbseq 0):       fl_fbseq_4/0.0 (fbseq 0):       fl_fbseq_4/0.0
 (fbseq 0):       fl_fbseq_5/500 (fbseq 0):       fl_fbseq_5/500
 (fbseq 1): txArray_fbseq_1/FB5 (fbseq 1): txArray_fbseq_1/FB5
 (fbseq 1):       fl_fbseq_2/220 (fbseq 1):       fl_fbseq_2/220

 ********** WARNING **********  (fbseq 1):       fl_fbseq_3/5.5 no match
```

Fig. 7: Comparison protocol of the plausibility control

In fig. 7 an example comparison protocol of the plausibility control with regard to two redundant FD, FD 1 and FD 2, is presented. There are appropriate tools for the automatic creation of such a comparison protocol. At first the parameter values of FD 1 are compared with FD 2 and finally the parameter values of FD 2 with FD 1. Both comparisons yield two warnings. Evidently both FD differ in one parameter value (see "Warnings": parameter $25.5 \neq 5.5$). Whether this is an error or an intentional deviation has to be clarified with the developer of the FD, of course. Thus the plausibility control generates hints at potential specification errors in redundant FD.

Summary

RETRANS was developed on a HP workstation under HP-UX Release 9.05. At the moment the tool runs under HP-UX 10.20. The source code of the RETRANS consists of about 9000 lines of code. The amount of run-time for the analysis already conducted has been between 45 h and 75 h.

The tool RETRANS distinguishes 129 inconsistencies during the analysis of the FDG modules and 100 inconsistencies during the analysis of the FD modules.

RETRANS has been tested in detail with extensive data including such from real applications in nuclear power plants.

According to the state of the art, national and international standards require the verification of each step in the software development process (see fig. 1). Of course the development step of automatic code generation has to be verified too. RETRANS assists in performing the required verification step of the application software that is used for the realisation of I&C functions in TELEPERM XS systems for safety relevant applications completely and reliable. On the contrary to alternative assessment techniques as the manual code analysis/inspection the tool-based assessment offers - with complete verification of the source code against the underlying specification - not only a considerable reduction of effort (cost advantage) but also avoids human factor problems which will appear during manual verification of extensive source codes.

The application of RETRANS is foreseen for the new digital I&C systems of the nuclear power plants FRM-II, Paks and Bohunice. Further, its usage for the new digital I&C technique system of Beznau is planned.

References

/1/ Graf, A., "Software Development Method for Safety Critical Applications", in: Haapanen P. (ed) Advanced Control and Instrumentation Systems in Nuclear Power Plants, Design, Verification and Validation, Proc. of TQM Espoo/Helsinki, Finland, 20 - 23 June 1994.

/2/ Miedl, H., "Reverse Transformation of Normed Source Code : Development of a Tool to Demonstrate the Functional Equivalence of Normed Source Code with its Specification", Proc. of Probabilistic Safety Assessment and Management '96, ESREL '96 and PSAM-III, Crete, Greece, 24 - 28 June 1996.

/3/ Brummer, J., Kersken, M., Lindner, A., Miedl, H., "Validation of Tranformation Tools", Licensing of Computer-Based Systems Important to Safety, NEA/CNRA/R(97)2, p. 400 - 411.

/4/ Lindner, A., Miedl, H., "Methodology and Tools for Independent Verification and Validation of Computerised I&C Systems Important to Safety, Specialists Meeting on Computerised Protection and Safety Related Systems in Nuclear Power Plants (IAEA/IWG-NPPCI), Budapest, Hungary, 27 - 29 October 1997

NEXT PAGE(S) left BLANK