Adv Syst Sci Appl 2021; **01**:113–138
Published online at https://ijassa.ipu.ru.

Software for Testability Analysis of Aviation Systems

Valentina S. Viktorova[1]*, Armen S. Stepanyants[1]

[1] *V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, Russia*

Abstract: This paper describes models, methods and software tool for testability analysis of aviation systems. It comprises analytical and programing aspects of calculations of main testability, reliability and availability indices. It presents general description of the software, XML schema of input data and technique of their mapping to the database structure. The procedure for generating the initial data for testability analysis based on the line replaceable units failure modes report is described. Fault tree model for analysis of the built-in test conformity is suggested. Markov models have been created for analyzing reliability and availability, taking into account the features of the built-in test and the specifics of the aircraft operation. An approach to the construction of trends in the operative availability of aviation systems in the inter-maintenance interval is proposed.

Keywords: testability, built-in-test, fault coverage, failure detection ratio, built-in-test equipment conformity, failure mode and effects analysis, Markov reliability models, operative availability trend

1. INTRODUCTION

Reliability, availability, testability, maintainability are the main properties of aviation items that determine the safety of an aircraft flight [1]. These properties are interrelated (Chapter 6 in [2]), an improvement (worsening) of one leads to a corresponding changes in the others. Testability is a property of an item that characterizes its adaptability to be controlled by specified diagnostic systems. These systems include both internal and external diagnostic means. Internal diagnostic means integrated on-line referred to as built-in-test (BIT). In one of the first regulatory documents in this area [3], the targets of testability analysis are defined as assessing the ability to detect and isolate system faults to the faulty replaceable assembly level. Our research focuses on design stage advanced testability analysis, including investigation of impact of BIT performance on the reliability and availability of aircraft systems. Analytical models that consider imperfect BIT performance are known as imperfect fault coverage models. Many articles are devoted to these models and associated reliability analysis techniques [4–8]. We have developed a reliability model considering imperfect fault coverage and aircraft-specific recovery strategies for the latent failures. When constructing the reliability models, a technical inspection and maintenance program [9, 10] typical for civil aviation was taken into account.

At the aircraft design stage, it is mandatory to carrying out a qualitative and preliminary quantitative failure modes and effects analysis (FMEA) [11]. If the FMEA table records are supplemented with information on the failure detection methods [13–15], then they can be used as input data for calculating the testability indices and conducting reliability and availability analysis taking into account the BIT characteristics. Testability-oriented FMEA

*Corresponding author: vsviktorova@gmail.com

analysis is performed for all functional aircraft systems on the level of the line replaceable units (LRU) [16]. Storage, selection, sorting and filtering of data of such complexity and volume as aircraft FMEA reports can be carried out only with the involvement of modern database management systems. Implementation of code of definition and solving reliability and testability analysis models is only possible through the advanced programming languages with high precision numeric data types and support of dynamic data structures. We chose Oracle DBMS and PL/SQL programming language.

In this paper, we present software for joint testability and reliability analysis of civil aircrafts. The models and analysis methods used in this software reflect the experience we gained during the project testability analysis of the Sukhoi Superjet and MC-21 aircraft family. This software provides the following options

- testability indices calculation;
- construction of distributions of testability indices by levels of criticality of failures;
- construction of distributions of testability indices by fault detection methods and by impact on departure delay;
- evaluation of built-in-test equipment conformity;
- automatic construction of Markov reliability models of redundant LRU assemblies with imperfect fault coverage and aviation-specific recovery scenarios;
- investigation of the impact of BIT functional features on system reliability and availability;
- creation of operative availability trends within the specified time frame between the aircraft maintenance periods.

The rest of this paper is organized as follows. In Section 2, we give the software description including meta definition of testability data and their mapping to the database structure. Section 3 presents mathematical background of the testability&reliability analysis. In section 3.1, we derive the formulas for calculating single testability measures. In section 3.2, we define logical-probability model of BIT conformity. In section 3.3, we suggest the parametric Markov reliability model of redundant LRU assemblies with imperfect fault coverage by BIT. Section 3.4 describes an approach to trending of aircraft operative availability. Some results and graphical illustrations of the analysis are summarized in Section 4. In the appendix, we provide the XML Schema of testability data.

2. TESTABILITY ANALYSIS SOFTWARE DESCRIPTION

2.1. Testability metadata structure

The actual practice of database design is the preliminary creation of a platform-independent meta-definition of the data. In accordance with the recommendations of international reference books [17–19] on technical publications in aerospace industries, the XML language is used for the meta definition of information items of this software. Special language referred to as XML Schema Definition (XSD) [20] are used for description of the structure, data types, elements and attributes of an XML document. We have developed the XML Schema that defines the XML image of testability data - Testability Data XML Schema Definition (TXSD). The pilot version of the TXSD was described in [21].

Current TXSD is presented in Appendix. It defines four-level hierarchical structure with the help of four Global Complex Types - TProjects, TSystems, TLRUs, TFMEAs. Each of them is an unbounded sequence of elements of type TProject, TSystem, TLRU, TFMEA, respectively. The complex type TProject defines project. Project is a generic term. An aircraft as a whole, arbitrary functional systems, for example, air conditioning, fire protection, integrated modular avionics can be considered as a project. Subsystems and assembles, for example, a landing gear steering or an engine interface unit of the central computer module of the indicating/recording system also may be represented as a project. Thus, the level of

detail of the analyzed object is determined by the user. The TProject context is divided into three parts - descriptors (codes and names of the project), time parameters (maintenance periods, recovery times...), set of functional systems (TSystems). The TSystem context also is divided into three parts - descriptors (codes and names of the system), technical specifications for the system reliability and testability indices, set of the line replaceable units (TLRUs). The TLRU context is divided into six parts - descriptors (codes and names of the LRU), LRU part and serial numbers, LRU identifier in the centralized maintenance system (CMS), redundancy schema, technical specifications for the reliability and maintainability indices, set of failure entries (TFMEAs). The TFMEA type describes output data of design stage analysis of failure modes and effects of the LRU. TFMEA has complex structure. The elements of this structure define various properties of failure, including failure descriptors in CMS, failure rate, failure severity level, detection method, phase of flight, flag of departure delay, BIT resolution features and so on.

TXSD contains Global Simple Types which define

- string type of ATA code [22] (TCodATA) of system and LRU;
- decimal types of time parameters and reliability indices (TRIndex), normalized testability indices (TRatio);
- enumeration types of LRU properties - LRU redundancy schema (TRSchema), LRU mode (TLRUType);
- enumeration types of failure properties - detection method (TFDM), severity level (TSeverity), flight profile [23] (TPhases), mode (TLRUFailure), departure delay flag (TDelay).

TXSD includes a specialized complex datatype, namely resolution list (TLRUchain). It is an unbounded sequence of TCodATA elements. TLRUchain defines a list of replaceable units that will be isolated by BIT when a failure is detected.

A global element named ”Projects” specifies the root of any TXSD-based XML document. The complete composition of the TXSD is given in the Appendix.

2.2. Testability database structure

The mapping of TXSD to a relational database schema is carried out according to the following rules:

- global complex types TProject, TSystem, TLRU, TFMEA define the presence and structure of four main database tables storing information about projects, functional systems, LRU, FMEA;
- simple type elements define the data types, identity, check constraints of columns of these tables;
- elements of the sequence type TSystems, TLRUs, TFmeas contained in complex elements Project, System, LRU respectively define referential integrity within the database;
- nesting order of TXSD elements determines relations by foreign key between parents and childes tables of the database;
- enumeration type elements are embodied in look-up, reference tables.

Fig.2.1 presents graphical interpretation of the mapping procedure.

2.3. Testability software features

Testability analysis software is a Web application that runs on an Oracle database (Fig. 2.2). This application is built using Oracle database’s native low-code development platform APEX. This choice enables to build high-performance, scalable, secure apps. Oracle APEX uses a 3-tier architecture where requests are sent from the browser, through a web server, to the database. Within the database, the request is processed by Oracle APEX. Once the

processing is complete, the result is sent back to the browser. Processes of requesting and submitting pages are realized through a modern implementation of the Oracle net listener called Oracle Rest Data Service (ORDS). User interface of the software is mobile friendly. Pages have smart layout and include forms, charts, reports and other user interface components that can work across varying screen resolutions.

Database includes main tables (see subsection 2.2), reference tables and views. Views contains data from join of FMEA and reference tables. Specifications and bodies of analytical procedures and functions are contained in four PL/SQL packages - TestPckg, RelPckg, APckg, NumPckg. TestPckg, RelPckg, APckg includes procedures and functions of testability, reliability, availability analysis respectively. NumPckg implements numerical methods for mathematical computations .

Reports of testability and reliability analysis can be download into external files of csv, xlsx, rtf, xml, pdf formats.

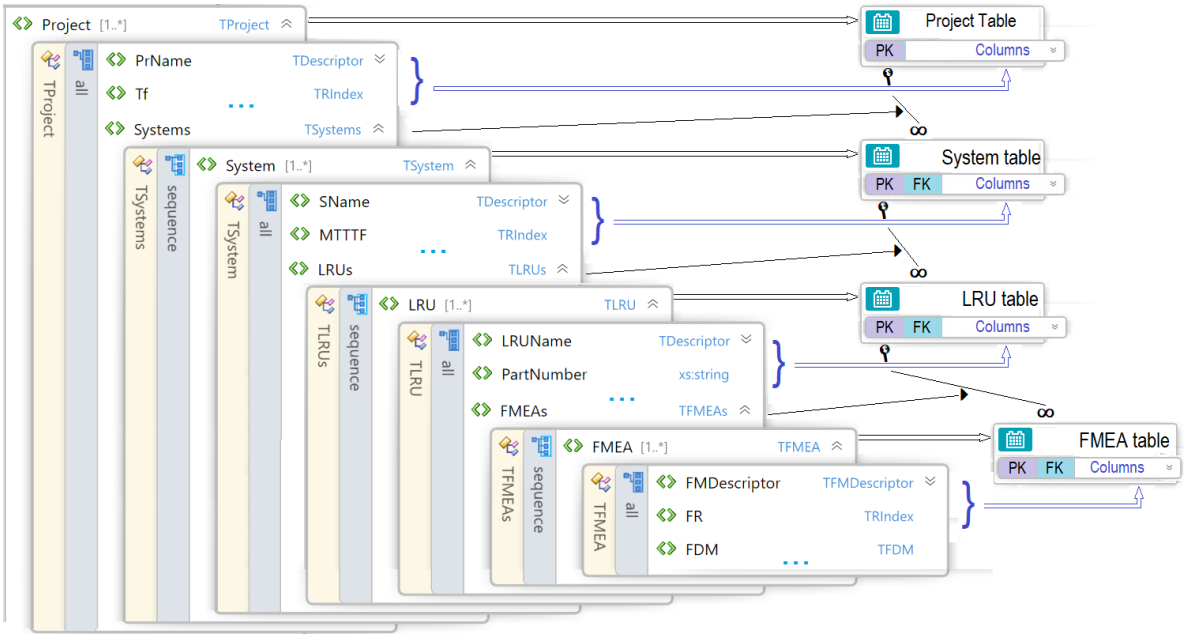

Fig. 2.1. TXSD to database schema conversion (FK – foreign key, PK – primary key).

.

3. MATHEMATICAL BACKGROUND OF COMPUTATIONAL ALGORITHMS

The following subsections describe models, methods and calculation procedures applied in the software for quantification and analysis of various testability, reliability and availability indicators. We use the generally accepted in aviation reliability analysis assumption about an exponential distribution of random times to failure [12]. In this case, the reliability indicator called the failure rate is equal to the constant parameter of the distribution.

3.1. Single testability indicators

Single testability indicators (STI) are used to specify BIT functionality or reliability. These indicators are used to assess a single BIT property. When calculating STI, a preliminary partition of the entire set of the LRUs failure modes into disjoint subsets is carried out.

STI can obviously be estimated as the ratio of the number of selected failures modes (N_s) to the total number (N_T) of failure modes

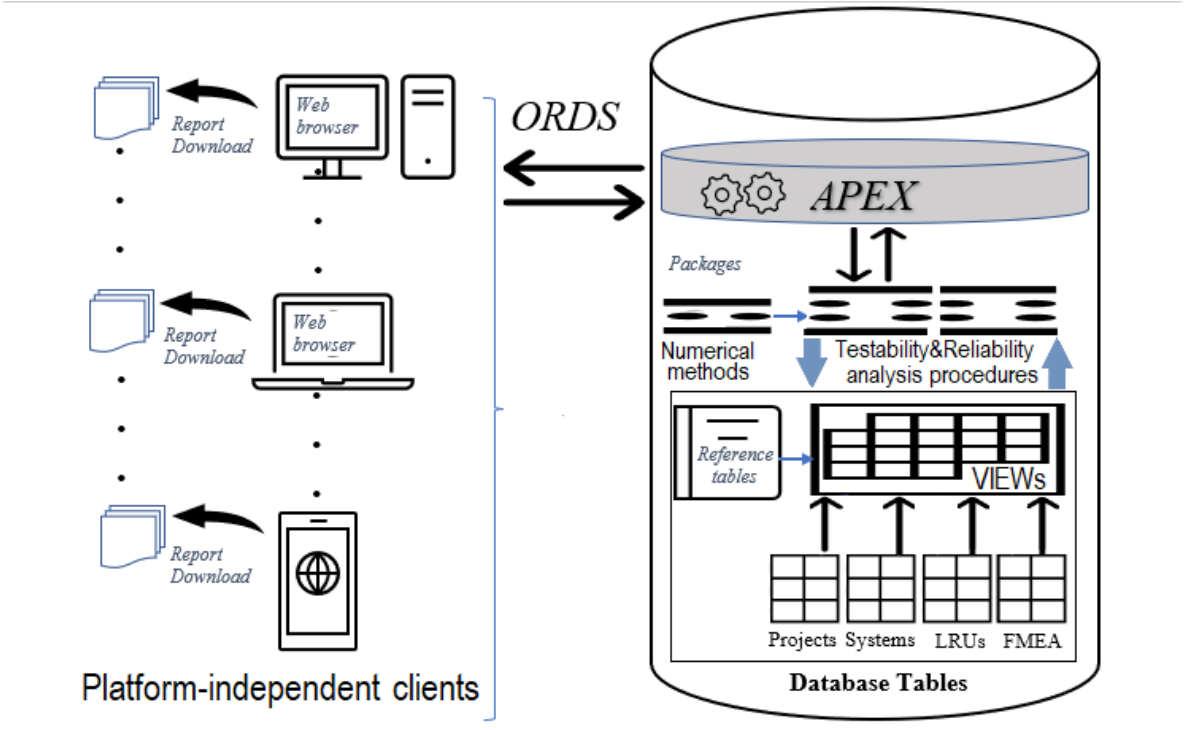

Fig. 2.2. Testability analysis software structure.

.

$$STI = \frac{N_s}{N_T}. \tag{3.1}$$

However, for joint reliability modeling of the object under test and BIT, STI should be set as the ratio of some probabilistic measures. The expediency of such definition is explained by the fact that when modeling reliability behavior, it will be possible to split the total flow of the object failures into several components interesting for researcher. For example, detected failures and hidden (latent) failures. In this case, STI can be defined as the conditional probability of the failure modes of interest, provided that the failure of the object has occurred:

$$STI = \frac{(1-\exp^{-\tilde{\Lambda}_S t})}{(1-\exp^{-\tilde{\Lambda}_T t})}, \tag{3.2}$$

where $\tilde{\Lambda}_T = \frac{1}{t}\int_0^t \Lambda_T(\tau)d\tau$ – average failure rate of the object; $\tilde{\Lambda}_S = \frac{1}{t}\int_0^t \Lambda_S(\tau)d\tau$ – average rate of selected failure modes of the object.

If we consider the exponential distribution of random time to failure, then

$$STI = \frac{(1-\exp^{-\Lambda_S t})}{(1-\exp^{-\Lambda_T t})} \underset{\lambda t \ll 1}{\approx} \frac{\Lambda_S}{\Lambda_T}. \tag{3.3}$$

If we assume the incompatibility of the object failure modes

$$STI = \frac{\frac{\Lambda_S}{\Lambda_T}(1-\exp^{-\Lambda_T t})}{1-\exp^{-\Lambda_T t}} = \frac{\Lambda_S}{\Lambda_T}. \tag{3.4}$$

Most testability indicators given in reference books [1, 3] are the ratios similar to (3.1), (3.4).

The main single testability indices are:

- fault detection ratio (FDR);
- fault isolation ratio (FIR);
- BIT efficiency;
- BIT unreliability factor (UF);
- BIT false alarm factor (FAF).

They are formed based on summing the failure rates (failure numbers) fetched from the FMEA table: $\sum_{i\in\Omega_1}\lambda_i/\sum_{j\in\Omega_2}\lambda_j$ or $\sum_{i\in\Omega_1}N_i/\sum_{j\in\Omega_2}N_j$. The software executes SQL queries to FMEA table and generates the following subsets of the set of failure modes:

- Ω – total set of the object failures;
- Ω_b, Ω_m, Ω_c – disjoint subsets of the object failures detected by BIT, maintenance, crew, respectively (see Appendix, TFDM type);
- Ω_n – subset of the object hidden (latent) failures;
- Ω_{b1} – subset of the object failures detected by BIT and isolated to single faulty LRU;
- Ω_I, Ω_{II}, Ω_{III}, Ω_{IV}, Ω_V – disjoint subsets of the object failures related to severity levels I, II, III, IV, V, respectively (see Appendix, TSeverity type);
- Ω_{bI}, Ω_{bII}, Ω_{bIII}, Ω_{bIV}, Ω_{bV} – disjoint subsets of the object failures detected by BIT and related to severity levels I, II, III, IV, V, respectively;
- Ω_{nop}, Ω_{fa} – disjoint subsets of BIT failures of type no operation and false alarm (see Appendix, TFailureType);
- Ω_{ff}– subset of functional failures;
- Ω_d– subset of failures impacting on departure delay (see Appendix, TDelay type).

Table 3.1 shows how sets of failure modes Ω_1, Ω_2 are formed when calculating single testability indices.

Table 3.1. Definition of $\Omega_1,\ \Omega_2$

index	Ω_1	Ω_2
Fault detection ratio	Ω_b	Ω
Fault isolation ratio	Ω_{b1}	Ω_b
FDR in flight	$\Omega_b \cup \Omega_c$	Ω
BIT efficiency	Ω_b	$\Omega_b \cup \Omega_m \cup \Omega_c$
BIT unreliability factor (UF)	$\Omega_{nop} \cup \Omega_{fa}$	Ω
BIT false alarm factor (FAF)	$\Omega_{fa} \cup \Omega_{fa}$	$\Omega_{nop} \cup \Omega_{fa}$
FDR for severity level i(FDR$_i$) (i=I,II,III,IV,V)	Ω_{bi}	Ω_i

Construction of distribution of fault detection ratio by levels of failures criticality are based on FDR$_i$ calculation.

In the reliability theory, in particular, when solving maintenance problems, the index of average failures number (AFN) is used. This indicator is the mathematical expectation of the number of the object failures during a given time interval $(N(t))$. At exponentially distributed with parameter λ random times to failures, AFN is calculated by the formula [2, pp. 89–90]:

$$N(t) = \lambda t. \tag{3.5}$$

AFN is not a regulated index of testability. But when it is calculated in relation to the failure modes covered by different detection methods, it becomes an obvious measure of testability. In this software, the AFN is calculated over flight and different maintenance periods.

3.2. BIT conformity

For an integral assessment of BIT quality we use a composed probabilistic indicator called the conformity of BIT. BIT conformity is a complex characteristic that takes into account both functional and reliability properties of BIT. In this case, two incompatible types of BIT failures are considered - non-operation and false alarm.

Let us derive an expression for BIT conformity under the following initial data and assumptions:

- event A means the controlled object is in operable (good) state ; event $\overline{A}$ means the controlled object is in inoperable (failure) state; $Prob(A) = P_A$, $Prob(\overline{A}) = Q_A$ are the probabilities of these events;
- BIT has two failure modes, the first mode is non-operation (I) and the second one is false alarm (II); event K is a good (failure-free) BIT state, event $\overline{K}_{NO}$ is occurrence of non-operation BIT failure, event $\overline{K}_{FA}$ is occurrence of false alarm BIT failure; $Prob(K) = P_K$, $Prob(\overline{K}_{NO}) = Q_{NO}$, $Prob(\overline{K}_{FA}) = Q_{FA}$ are the probabilities of occurrence of corresponding events;
- events K, $\overline{K}_{NO}$, $\overline{K}_{FA}$ are a complete group of mutually exclusive events, so $P_Q + P_{NO} + Q_{FA} = 1$;
- B is the event that BIT recognizes the object state as operable, $\overline{B}$ - BIT recognizes the object state as failed;
- η is a fault detection ratio of BIT.

We define the event of BIT conformity C as the correct recognition of the state of the object

$$C \;=\; (A \wedge B) \vee (\overline{A} \wedge \overline{B}). \tag{3.6}$$

To find the probability of a truth of this event $Prob\{C = 1\}$, we use the fault tree model. One of the techniques for constructing fault trees is to apply the theorem of decomposition into inconsistent hypotheses in some logical variables, which ensures the decomposition of the logical model and simplifies the construction of individual tree branches [24, pp. 165–172], [25, pp. 103–115]. To construct a fault tree with a top event C, we use the decomposition with respect to the object states (events A and $\overline{A}$). And in the tree branches with $\overline{A}$ we apply the decomposition with respect to BIT coverage, viz. insertion the inhibit gates with conditional events η and $1 - \eta$. The inhibit gate is used to indicate that the output occurs when the input events (bottom events) occur and the input condition (right side event) is satisfied. Fig. 3.3 shows the resulting fault tree. The fault tree contains repeated basic events (Event 1 ($\overline{A}$), Event 2 ($\overline{K}$), Event 4 ($\overline{K}_{FA}$)). Repeated events are colored green. The logical expression corresponding to the constructed fault tree model is given below.

$$\begin{aligned} C \;=\; A \wedge (K \vee \overline{K}_{NO}) \vee \overline{A} \wedge \eta \wedge (K \vee \overline{K}_{FA}) \vee \overline{A} \wedge (1-\eta) \wedge \overline{K}_{FA} = \\ A \wedge (K \vee \overline{K}_{NO}) \vee (\overline{A} \wedge \overline{K}_{FA}) \vee (\overline{A} \wedge \eta \wedge K). \end{aligned} \tag{3.7}$$

Let's write down the probability function for a truth of the found logical expression. Since the decomposition into inconsistent hypotheses was used when creating fault tree, we obtained logical expression in orthogonal form. Therefore, one can directly substitute the corresponding probabilities instead of logical variables, and replace logical operations with arithmetic ones [26, pp. 39,278–336]. The final expression for BIT conformity

$$Prob(C = 1) = P_C = P_A(P_K + Q_{NO}) + Q_A Q_{FA} + Q_A \eta P_K. \tag{3.8}$$

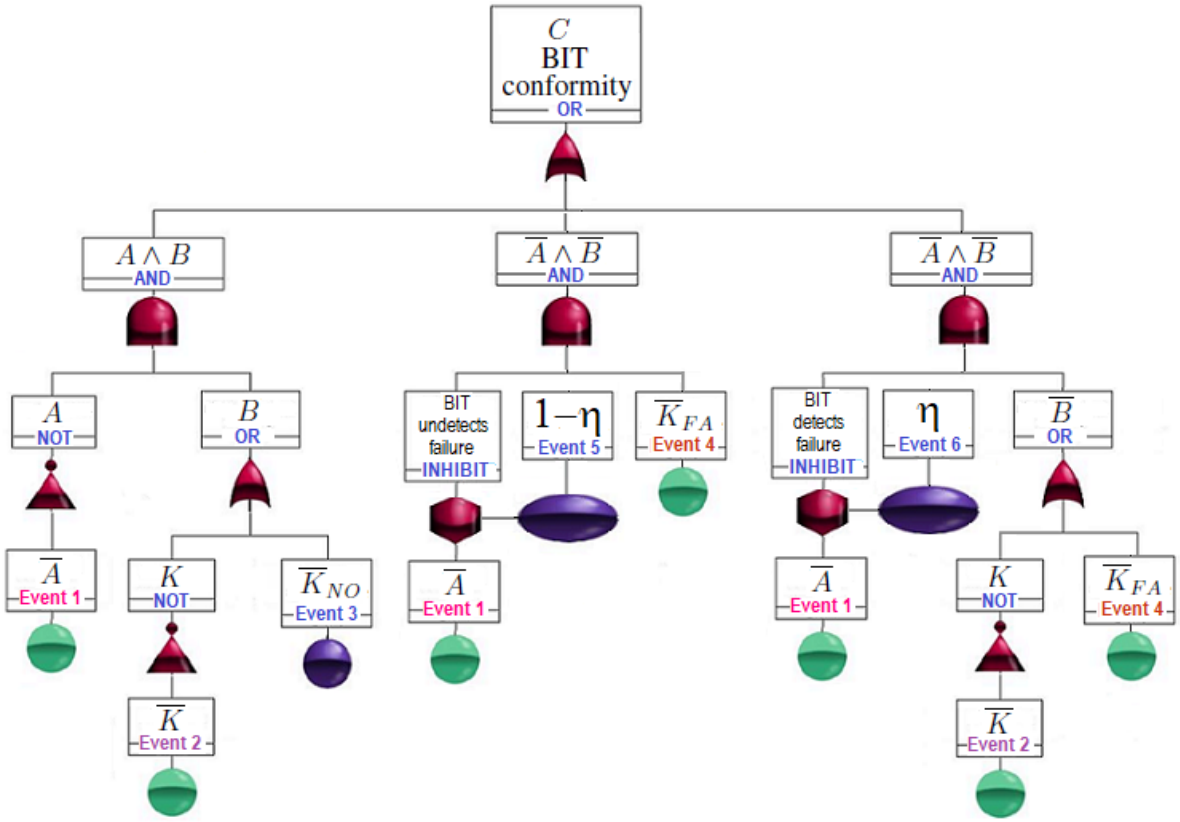

Fig. 3.3. Fault tree model of BIT conformity.

.

3.3. Reliability measures

Analytical modeling of the systems reliability behavior taking into account BIT FDR and redundancy is implemented on continuous-time Markov chains with discrete finite state space of size n. Markov reliability model (MRM) allows to model complex transient and steady-state reliability behavior and to estimate wide range of reliability and availability indices [25, pp. 147–199]. MRM is formalized by the system of Kolmogorov differential equations that describes the behavior of the state probability vector $P(t)$ as a function of time t:

$$dP(t)/dt \;=\; P(t)\Lambda;\; P(0) = P_0, \tag{3.9}$$

where $\Lambda \;=\; \|\lambda_{ij}\| - n \times n$ infinitesimal matrix, $\lambda_{ij}, i \neq j$ is the rate of transition from state i to state j, $\lambda_{ii} = -\sum_{i \neq j} \lambda_{ij}$. The i^{th} component $p_i(t)$ of the row vector $P(t)$ is the probability the system is in state i at time t.

The software performs automatic generation of the Markov model. The MRM describes reliability behavior of the redundant assembly of line replaceable units (RAL) between maintenance periods such as heavy checks. The model generation algorithm is based on the following provisions

- a set of non-redundant line replaceable units are combined into assembly based on functional or design features;
- both recovery and redundancy are implemented at the assembly level;
- redundancy type is parallel operating k out of m, where m - total quantity of identical assemblies, k - quantity required for success operation $(m \geq 1, k \leq m)$;

- incomplete coverage of LRU failures occurs due to functional imperfections in the BIT $(0 \leq \eta \leq 1)$;
- two RAL recovery scenarios between checks are considered
  - the RAL operability is restored in case of occurrence of the hidden failures of all assemblies (scenario I) (for example, engine control system);
  - the RAL operability is not restored in case of occurrence of the hidden failures of all assemblies (scenario II) (for example, a fire protection system).
- the presence of at least one failure detected by the BIT in the assembly leads to a complete assembly recovery, regardless of whether there are other (including hidden) failures in the assembly.

When building the model, the states are aggregated. All detected (hidden) failures of the assembly elements are combined into one state of a detected (hidden) assembly failure. Each MRM state is encoded with a m-character code $d_1 d_2 \ldots d_i \ldots d_m$. The symbol d_i of the code is determined by the rule

$$d_i = \begin{cases} 1 & \text{if the assembly is operable} \\ D & \text{if a failure detected by the BIT has occurred in the assembly} \\ H & \text{if a failure undetected by the BIT (hidden) has occurred in the assembly} \\ HD(DH) & \text{if both detected and hidden failures have occurred in the assembly} \end{cases}$$

To ensure the correctness of the aggregated MRM, it is necessary to operate with the mean recovery times of the assembly

$$MTTR_{assembly} = \frac{\sum_i \frac{\lambda_i}{\mu_i}}{\sum_i \lambda_i}, \tag{3.10}$$

where λ_i, μ_i are the failure and recovery rates of the i^{th} LRU of the assembly.

The MRM state set is partitioning into two subsets of operational (good) states Ω_G and non operational (failure) states Ω_F. The formal condition of belonging of the model state s_i to one of these subsets is

$$\begin{cases} s_i \in \Omega_G & \text{if total number of symbol "1" in state code} \geq k \\ s_i \in \Omega_F & \text{if total number of symbol "1" in state code} < k \end{cases}$$

The parameters of the procedure for generating the infinitesimal matrix of this MRM are m, k, η, recovery scenario number. Values of parameters m, k, η are determined based on SQL queries to the database. The Markov graph corresponding to the generated matrix at $m = 3, k = 1$, scenario I is shown in Fig. 3.4.

The software calculates on the MRM the following reliability measures

1. Interval measures
   Reliability $R(t)$ and Unreliability $U(t)$ on time interval $(0, T)$.
2. Point measure
   Availability $A(t)$ in time instant t.
3. Steady-state measure
   Mean time to first failure MTTFF which is defined on unlimited time interval $(0, \infty)$.

To calculate $R(t)$, it is necessary to solve the system (3.9), having previously made all failure states absorbing. This requires equating to zero the elements of rows of matrix Λ corresponding to failure states.

$$R(t) = \sum_{i \in \Omega_G} p_i(t), \quad U(t) = \sum_{j \in \Omega_F} p_j(t) = 1 - R(t). \tag{3.11}$$

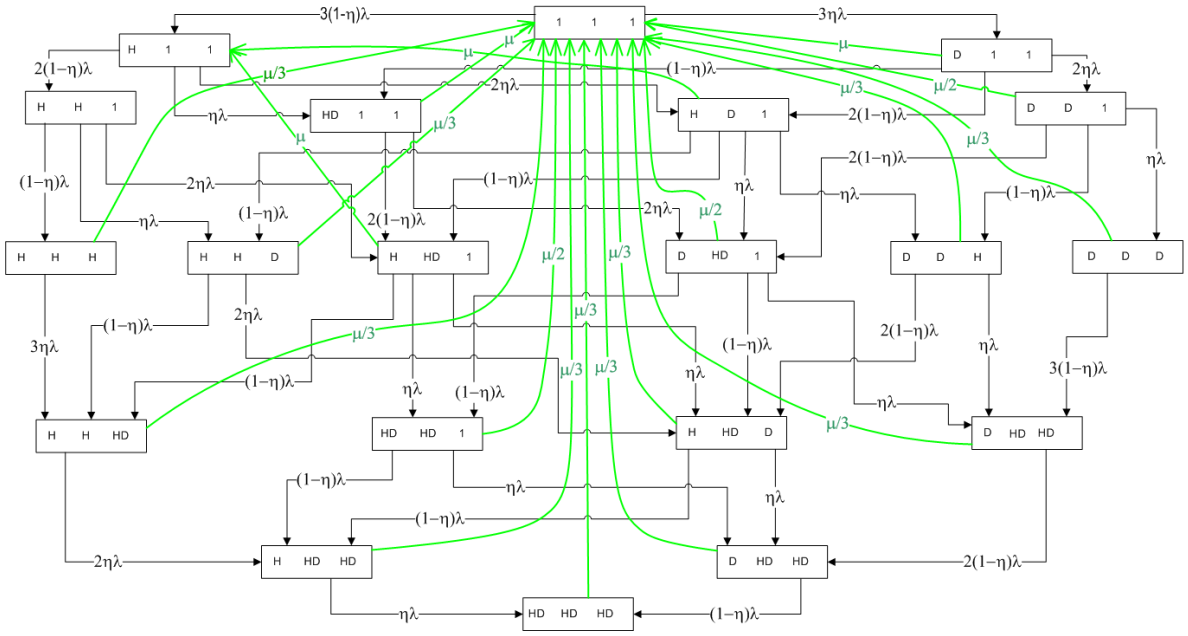

Fig. 3.4. Graph representation of the MRM ($m = 3, k = 1$, scenario I).

The calculation of $A(t)$ is performed in a similar way, but in this case the failure states should not be forced to be absorbing.

Calculation of MTTFF is reduced to solving a system of linear algebraic equations:

$$-P^*(0) = T\Lambda^*, \tag{3.12}$$

where $\Lambda^* - r \times r$ matrix derived from matrix Λ by deleting rows and columns corresponding to nonoperational states; r – total number of operational states $(r < n)$; $P^*(0) = [\ p_1(0)\ \ p_2(0)\ \ \ldots\ \ p_{r-1}(0)\ \ p_r(0)\ ]$. Provided that the system starts from a completely healthy state $P^*(0) = [\ 1\ \ 0\ \ \ldots\ \ 0\ \ 0\ ]$. MTTFF = $\sum_{i=1}^{r} T_i$.

The solutions of systems of differential (3.9) and algebraic (3.12) equations are implemented using numerical methods focused on the stiffness and bad conditioning [27], inherent in the reliability models of the repaired systems [28, pp. 48–59].

3.4. Operative availability model

Aviation systems are classified as hybrid recovery systems [29]. These systems have the cyclic mode of operation whose duty cycle consists of a target operation phases (TOP), repair phases (RP), inspection, maintenance and correction phases (IMCP). Recovery actions after failures are not possible on-line (TOP) and can be conducted only off-line during RP and IMCP. When the system is online (TOP) information about failures detected by the BIT is recorded in memory unit of CMCS (central maintenance computer system). CMCS collects and reports LRU faults data in order to aid maintenance personnel in recovery and maintenance procedures. During RP, only faults detected by the BIT can be eliminated. Full recovery of system operation is carried out only at stage IMCP where latent faults also eliminated. Thus, due to the imperfection of the built-in test, between IMCP stages there is a consistent decrease in the system availability at the beginning of each subsequent TOP. The main indicator that can be used to assess the reliability behavior of the cyclic mode systems, in particular aviation systems, is operative availability. The operative availability $A(t, t + t_0)$ is index depending on two variables – the operating time t (usually counted from 0), when recovery from failure states is possible, and the operative time t_0 (counted from t), when

recovery from failure states is impossible. In general, this index is calculated as

$$A(t, t+t_0) = \sum_{i \in \Omega_G} A_i(t) R(t, t+t_0)/_{S(t)=i}, \tag{3.13}$$

where $R(t, t+t_0)/_{S(t)=i}$ – the reliability of the system on the interval $(t, t+t_0)$, provided that the system is in state i at time t.

This software builds a trend of operative availability for a given number of phases of target operation (Q_{step}). The construction of the trend is carried out by a program that performs a step-by-step calculation of reliability $R(T_i)$ (T_i is the duration of the i^{th} TOP). The calculation is performed on the MRM shown in Fig. 3.5. A step-by-step calculation of the reliability $R(T_i)$ is carried out each time with new initial conditions, which makes it possible to take into account the fact of a decrease in availability arising from incomplete coverage of failures at the previous $i-1$ recovery phases (RP). The vector of initial conditions for the i^{th} TOP $C^{(i)}$ are formed according to the following rule

$$C^{(i)} = \left[\begin{array}{ccccccc} c_1^{(i)} & c_2^{(i)} & \dots & c_j^{(i)} & \dots & c_n^{(i)} \end{array} \right], \tag{3.14}$$

where

$$c_j^{(i)} = \begin{cases} p_1(T_{i-1}) + \sum_{k \in \Omega_b} p_k(T_{i-1}) & \text{if } j = 1 \\ p_j(T_{i-1}) & \text{if } j \in \Omega \backslash \Omega_b \\ 0 & \text{if } j \in \Omega_b \end{cases}$$

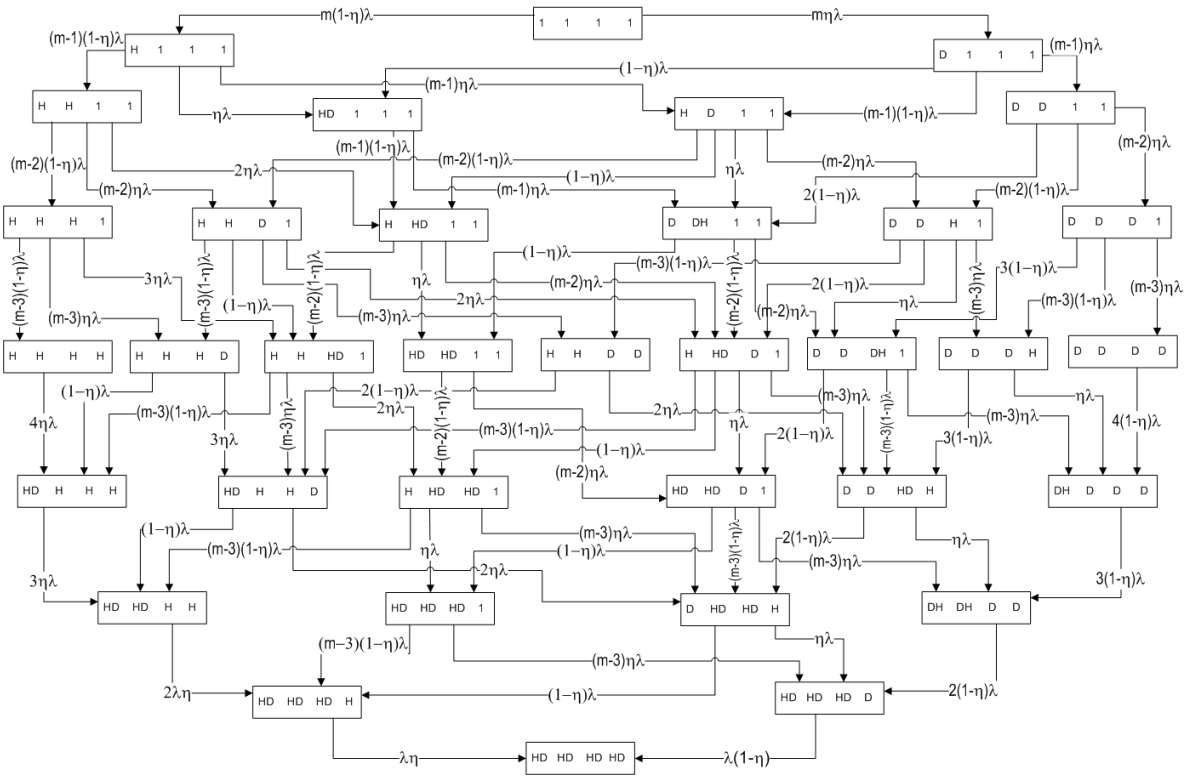

Fig. 3.5. Graph representation of the MRM over target operation phase ($1 \le m \le 4, 1 \le k \le m$).

4. CASE STUDIES OF THE SOFTWARE

Six cases below illustrates the software capability. The initial data when performing the calculations of testability and reliability indices are the output of the design stage analysis of modes and effects of LRU failures. A constant failure rate is specified in FMEA tables for each type of failures. The failure rates are calculated using the methods described in various reference books and standards [30] or/and processing statistics on failures. The calculation of interval indicators of testability and reliability are carried out for the periods of time regulated by the scheduling maintenance program referred as A B C D checks [10, pp. 75–84]. The complexity of service tasks grows from check A to check D. The actual periods of these inspections depend on the type of aircraft, but the lighter checks A and B are carried out more often than the heavier checks C and D. We use the following times between checks: $T_{CheckA} = 600\text{h}, T_{CheckB} = 3500\text{h}, T_{CheckC} = 7000\text{h}, T_{CheckD} = 70000\text{h}$. Flight duration $T_{Flight} = 5\text{h}$.

We demonstrate software capability on Sample System consisting of three LRU. Screenshot of the software page with the FMEA report is in Fig. 4.6.

LRU name	Failure ID	Failure rate [1/h]	FDM	Severity	Failure type	BIT resolution	Flight delay
Sample LRU 1	FID11	2.50E-04	BIT	Minor (IV)	FF	1	No
	FID12	6.00E-05	BIT	Marginal (III)	FF	1	No
	FID13	5.00E-05	BIT	Critical (II)	FF	1	Yes
	FID14	7.50E-04	None	Minor (IV)	FF	1	No
Sample LRU 2	FID21	5.00E-06	BIT	Catastrophic (I)	FF	1	Yes
	FID22	6.00E-05	Crew	Marginal (III)	BIT_FA	1	Yes
	FID23	5.00E-04	BIT	Minor (IV)	FF	1	No
	FID24	1.50E-03	None	Negligible (V)	FF	1	No
	FID25	1.50E-03	Maint	Negligible (V)	FF	1	No
Sample LRU 3	FID31	5.00E-06	BIT	Catastrophic (I)	FF	1	Yes
	FID32	5.00E-05	BIT	Critical (II)	FF	1	Yes
	FID33	6.00E-05	BIT	Marginal (III)	FF	1	Yes
	FID34	5.00E-04	Crew	Minor (IV)	BIT_FA	1	Yes
	FID35	1.00E-03	Maint	Minor (IV)	BIT_NOP	1	No
	FID36	2.00E-03	Maint	Negligible (V)	FF	1	No
	FID37	2.00E-03	BIT	Negligible (V)	FF	1	No

Fig. 4.6. Screenshot of the FMEA report.

4.1. Case 1. Failure modes classification

The software implements the classification of the LRU failure modes by following criteria:

- detection methods
  - BIT – failure is detected by built-in-test
  - MAINT – failure is detected in ground maintenance
  - CREW – failure is detected by crew
  - NONE – failure is not detected until Check D
- failure types
  - FF – functional failure
  - BIT NOP – no operation built-in-test failure
  - BIT FA – false alarm built-in-test failure
- impact on departure delay

The percentage of each group is calculated. The percentage is calculated by summing failure rates (estimation by frequency) Fig. 4.7, 4.8, 4.9 (a), or by the number of failures (quantification) Fig. 4.7, 4.8, 4.9 (b).

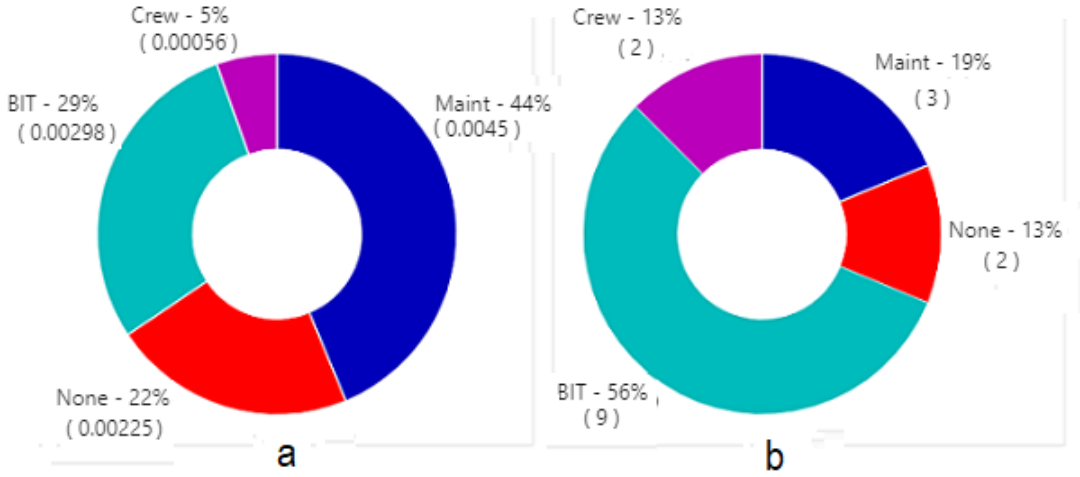

Fig. 4.7. Failure detection methods. a - estimate by frequency, b - quantification.

4.2. Case 2. STI calculation

The software implements calculation of the following single testability indices:

- total fault detection ratio including BIT, crew, maintenance (TFDR)
- BIT fault detection ratio (FDR)
- BIT fault isolation ratio (FIR)
- automated fault isolation capability ($AFIC = FDR \times FIR$)
- BIT efficiency

Calculations of TFDR, FDR, FIR, AFIC, BIT efficiency and creation of FDR distribution by failure mode severity levels are implemented in accordance with the definitions set out in section 3.1. Fig. 4.10 presents the screenshot with results of the analysis. Full coverage of high severity (catastrophic, critical) failures is desirable when designing BIT.

Comparison of frequency (I) and quantity (II) estimates can be useful in assessing the quality of testability solutions. For example, if the FDR I $<$ FDR II, then the design must be changed, since the most frequently failure modes are not covered by the BIT.

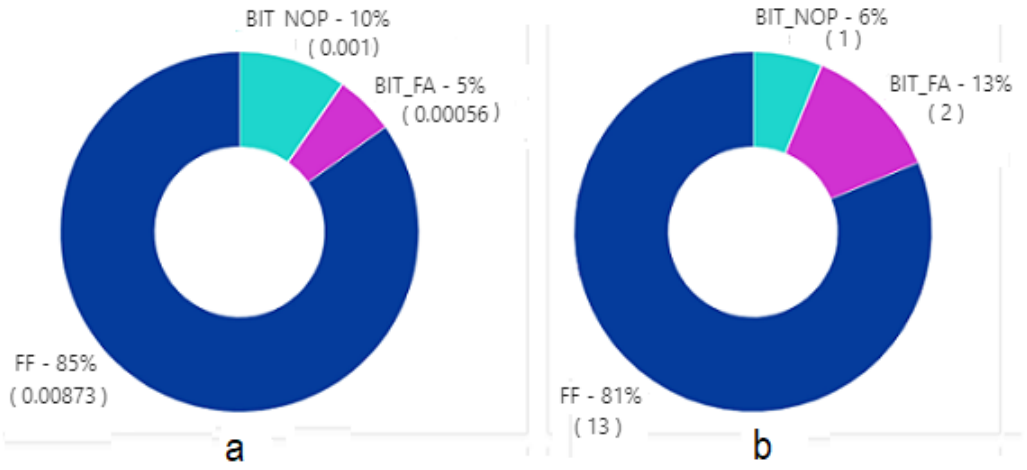

Fig. 4.8. Failure mode types. a - estimate by frequency, b - quantification.

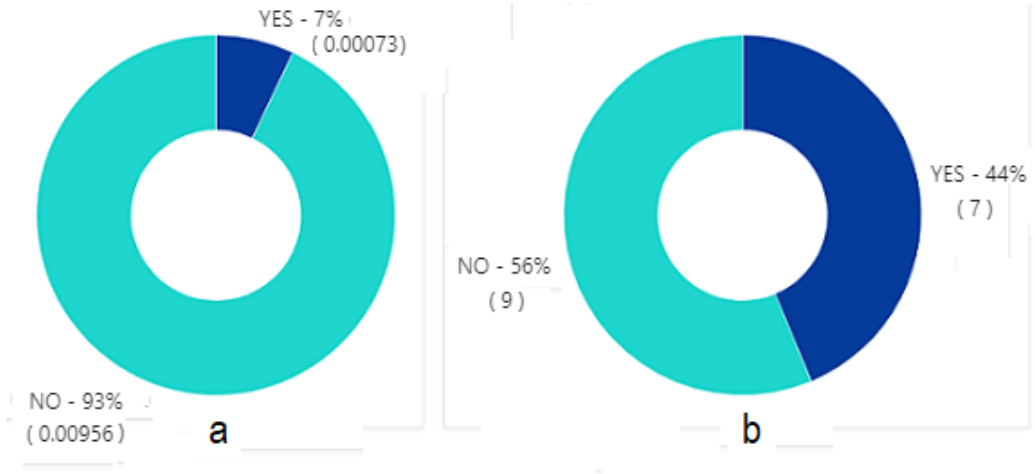

Fig. 4.9. Failures impact on departure delay. a - estimate by frequency, b - quantification.

4.3. Case 3. AFN calculation

The average failures number over time interval is calculated by (3.5) and the distribution of this indicator by failure severity is constructed. The calculation is performed for time intervals: flight, Check A, Check B, Check C. Distributions are constructed for the following types of failures:

- failures detected by built-in test (AFN BIT);
- failures detected during maintenance process (AFN MAINT);
- hiden (latent) failures not covered by BIT, crew and maintenance (AFN NONE).

Screenshots of the AFN summary page and respective charts are presented in Fig.4.11 and Fig.4.12.

4.4. Case 4. Analysis of BIT conformity

In the framework of BIT conformity analysis the following procedures are implemented:

- calculation of BIT conformity based on the technical specifications of BIT reliability and functional properties (Conformity I);

	fault detection ratio	fault detection ratio (BIT)	fault detection ratio (flight)	fault isolation ratio	automatic fault isolation capability (AFIC)	BIT efficiency %
Estimate by frequency	0.7813	0.2896	0.3440	1.0000	0.2896	37.06
Estimate by quantity	0.8750	0.5625	0.6875	1.0000	0.5625	64.29

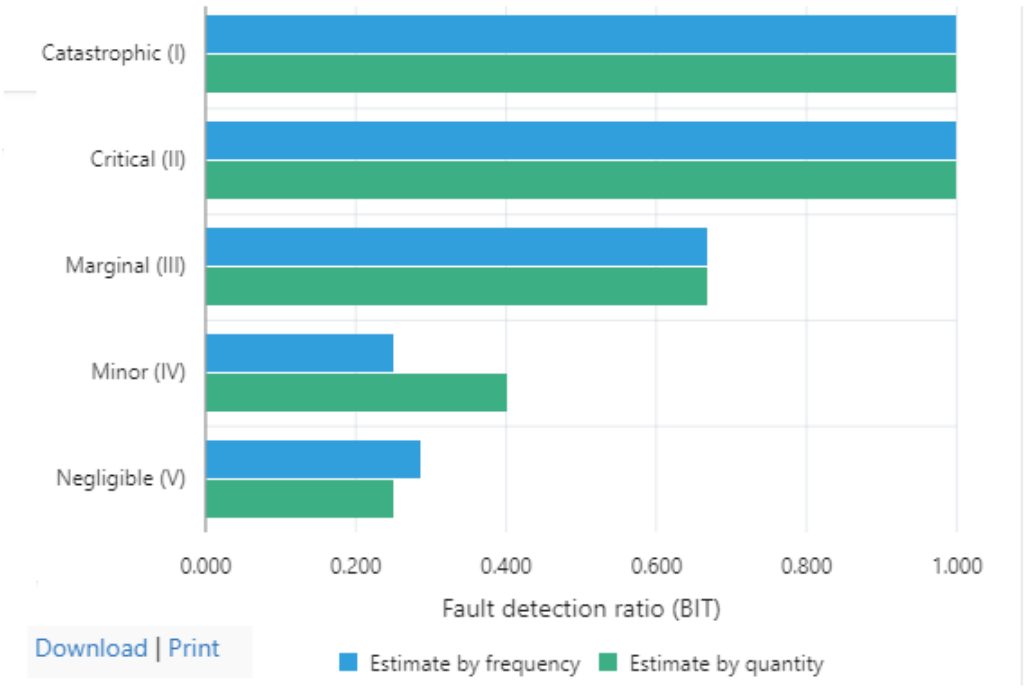

Fig. 4.10. Screenshot of the report of STI calculations.

.

- calculation of BIT conformity based on the FMEA data concerning rates and coverage of failures modes (Conformity II);
- comparison of the values of Conformity I and Conformity II and issuing recommendations for BIT modernization if Conformity II $<$ Conformity I;
- study of behavior of the conformity index in the space of BIT reliability parameters.

An approach to comparing identical indicators calculated on the basis of different type of initial data (technical specifications and FMEA) is proposed in [12]. Fig. 4.13 is a screenshot of the report of calculating BIT conformity I and II and their constituents.

The conformity indicator is calculated in accordance with the definitions set out in section 3.2. The plots in Fig. 4.14 (a) and (b) show the results of calculation of BIT conformity as a function of BIT unreliability factor UF (see table 3.1). UF characterizes unreliability of BIT equipment relative to unreliability of object under test. The set of curves is formed by varying the parameter k_{fa}. k_{fa} determines the ratio of the failure modes of BIT and is calculated as FAF (see table 3.1). Plot of Fig. 4.14 (a) shows small decrease of conformity appears linear

AFN summary table

Fault detection method	Failure severity	AFN Flight	AFN Check A	AFN Check B	AFN Check C
BIT	Catastrophic (I)	5.0E-05	6.0E-03	3.5E-02	7.0E-02
BIT	Critical (II)	5.0E-04	6.0E-02	3.5E-01	7.0E-01
BIT	Marginal (III)	6.0E-04	7.2E-02	4.2E-01	8.4E-01
BIT	Minor (IV)	3.8E-03	4.5E-01	2.6E+00	5.3E+00
BIT	Negligible (V)	1.0E-02	1.2E+00	7.0E+00	1.4E+01
Crew	Marginal (III)	3.0E-04	3.6E-02	2.1E-01	4.2E-01
Crew	Minor (IV)	2.5E-03	3.0E-01	1.8E+00	3.5E+00
Maint	Minor (IV)	5.0E-03	6.0E-01	3.5E+00	7.0E+00
Maint	Negligible (V)	1.8E-02	2.1E+00	1.2E+01	2.5E+01
None	Minor (IV)	3.8E-03	4.5E-01	2.6E+00	5.3E+00
None	Negligible (V)	7.5E-03	9.0E-01	5.3E+00	1.1E+01

Fig. 4.11. Screenshot of summary of the system average failures number.

in UF on short time interval (Flight). Plot of Fig. 4.14 (b) demonstrates limit behavior of conformity on long time interval (Check A). $(\lim_{t\to\infty} C(t) = k_{fa})$.

4.5. Case 5. Reliability analysis.

The calculation and study of reliability indicators are conducted on models described in section 3.3. The formation of the elements of matrix Λ is carried out by executing SQL-queries to FMEA table to select the failure modes rates and average recovery times of the LRU. The curves of reliability (unreliability), availability (unavailability), MTTFF are plotted as a function of FDR. A family of curves is plotted for each indicator. A separate curve corresponds to one of three standard redundancy schemes - duplicated (1 out of 2), tripled (1 out of 3), majority (2 out of 3). The fourth curve corresponds to a non-redundant assembly configuration (1 out of 1).

The chart in Fig. 4.15 (a) demonstrates a strong dependence of MTTFF on fault detection ratio. A sharp jump in the MTTFF is observed at FDR values close to 1. The chart in Fig. 4.15 (b) demonstrates a weak dependence of the Reliability on FDR at short time intervals. With an increase in the time interval, this dependence increases (Fig. 4.15 (c). Fig. 4.15 (c) also shows that a 2 out of 3 redundancy schema can only be used in conjunction with high fault detection ratio. Otherwise, this option turns out to be worse than the non-redundant scheme. On long time intervals, high BIT FDR and redundancy can dramatically increase the assembly Reliability (Fig. 4.15 (d)). Fig. 4.16 (a) and (b) show Availability as a function of FDR. The curves of this chart show that BIT FDR is a factor affecting the efficiency of using redundancy at recovery systems.

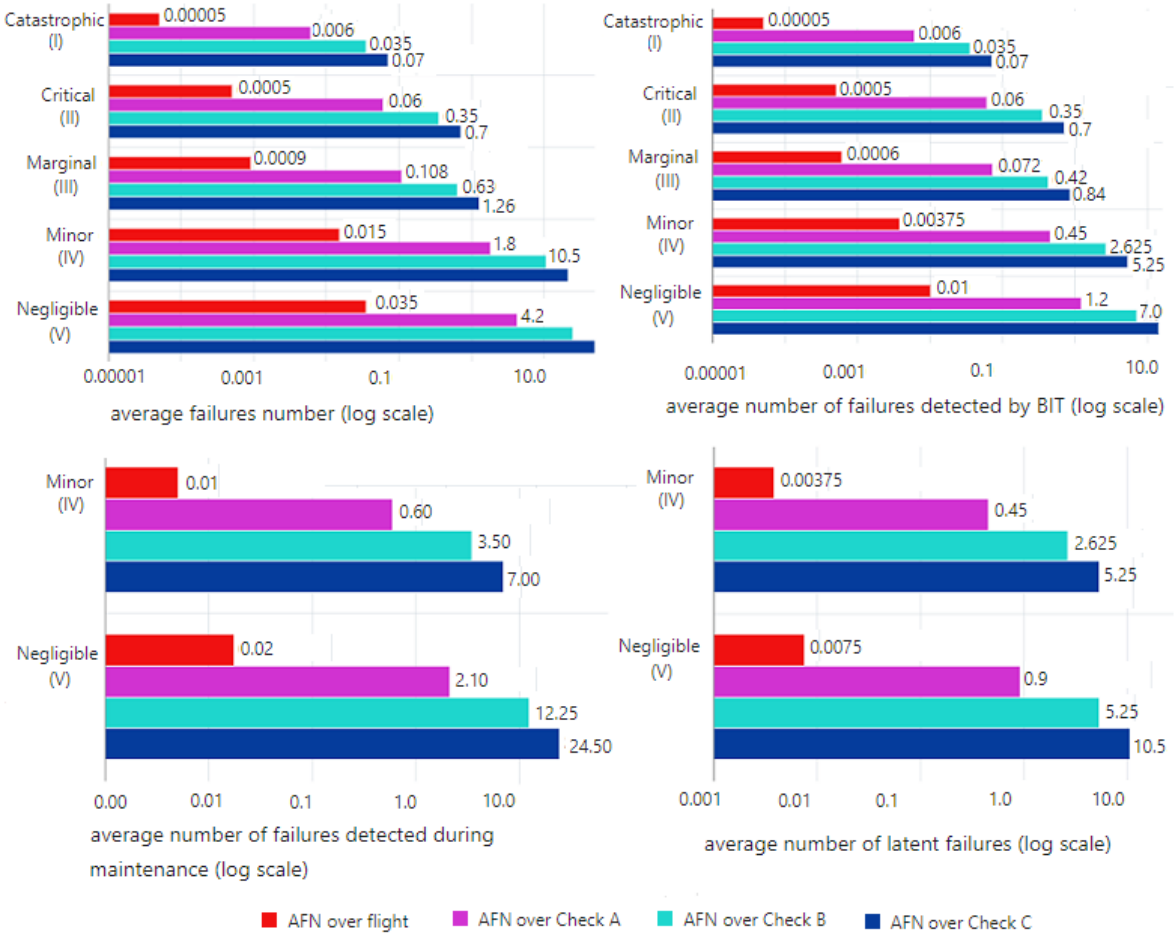

Fig. 4.12. Distribution of the system average failures number by severity and detection methods.

.

BIT Conformity over Flight (Tf), Check A (Ta)

Index	Requirements (Tf)	FMEA (Tf)	Requirements (Ta)	FMEA (Ta)
Mean Time to First Failure [h]	100.00	97.18	100.00	97.18
Unreliablity	4.8771E-02	5.0149E-02	9.9752E-01	9.9792E-01
Mean Time to First Failure BIT	1,000.00	641.03	1,000.00	641.03
Unreliability BIT	4.9875E-03	7.7697E-03	4.5119E-01	6.0781E-01
BIT unreliability factor	0.1000	0.1516	0.1000	0.1516
Fault detection ratio	0.7000	0.2896	0.7000	0.2896
Fault isolation ratio	0.9000	1.0000	0.9000	1.0000
BIT FA ratio	0.5000	0.3590	0.5000	0.3590
BIT NOP ratio	0.5000	0.6410	0.5000	0.6410
Probability of BIT FA	2.4938E-03	2.7891E-03	2.2559E-01	2.1819E-01
Probability of BIT NOP	2.4938E-03	4.9806E-03	2.2559E-01	3.8962E-01
BIT Conformity	9.7955E-01	9.6175E-01	5.7185E-01	3.3270E-01

Download | Print

Fig. 4.13. Screenshot of the BIT conformity analysis report (conformity I vs conformity II)

.

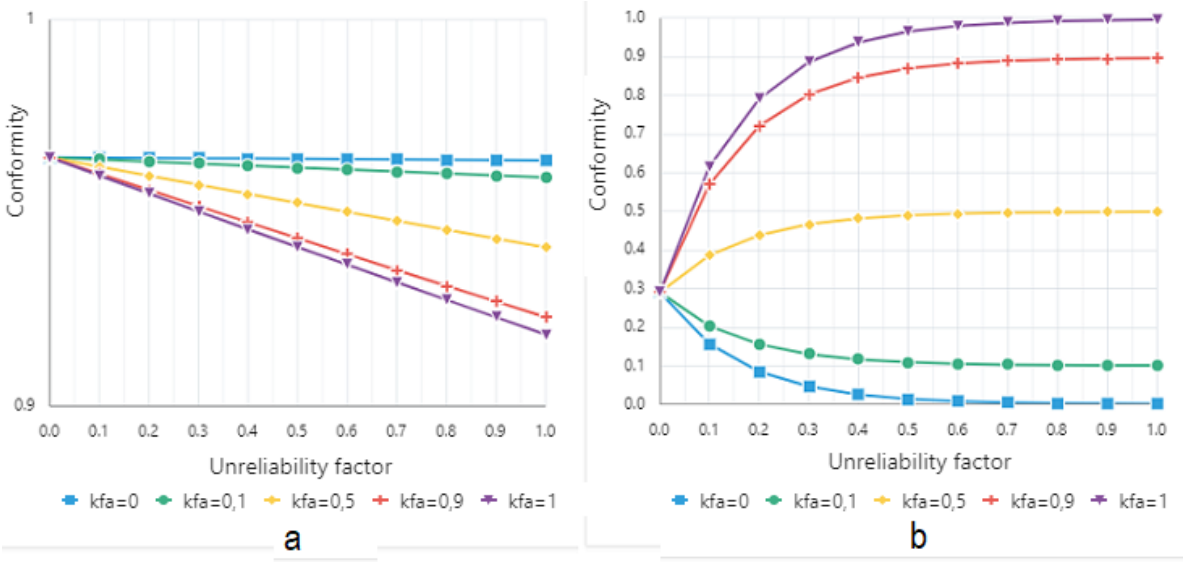

Fig. 4.14. BIT conformity vs unreliability factor. a. $t = T_{Flight}$, b. $t = T_{CheckA}$

.

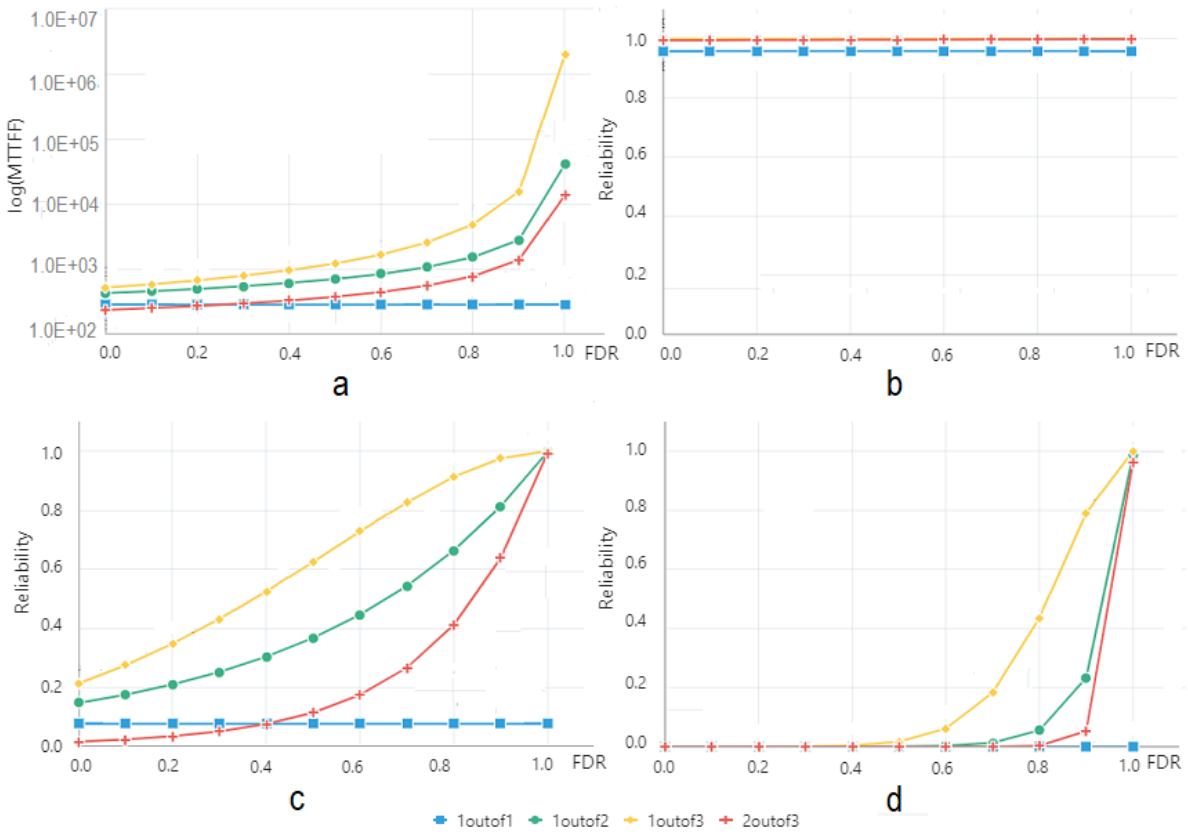

Fig. 4.15. Reliability indices vs BIT fault detection ratio
a. MTTFF b. $R(0, T_{Flight})$ c. $R(0, T_{CheckA})$ d. $R(0, T_{CheckB})$.

.

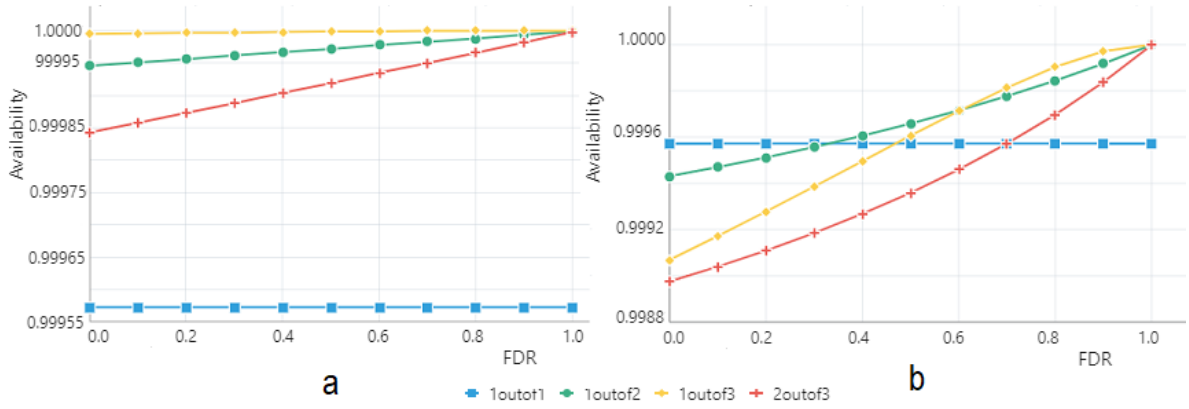

Fig. 4.16. Availability vs fault detection ratio a. $A(T_{Flight})$ b. $A(T_{CheckA})$

. .

4.6. Case 6. Operative availability analysis.

The software provides the ability to build an Operative Availability trend for an arbitrary time frame. The time frame is defined by the user by choosing the start time counted from the last maintenance cycle with 100% renewal and quantity of the TOPs. The trend construction is based on models of operative availability described in section 3.4. Trend analysis enables the aircraft designer to define BIT functionalities and maintenance periods during which latent failures are detected and eliminated. This avoids a decrease of the operative availability below the acceptable level. Fig. 4.17 (a,b) shows the trends in operative availability at FDR values equal to 0.7 and 0.9. The curves clearly show that an increase in the FDR leads to increase a probability of starting the system from a failure-free state. The limiting value of this probability, equal to one, is achieved at 100% failure coverage by BIT Fig. 4.17 (c).

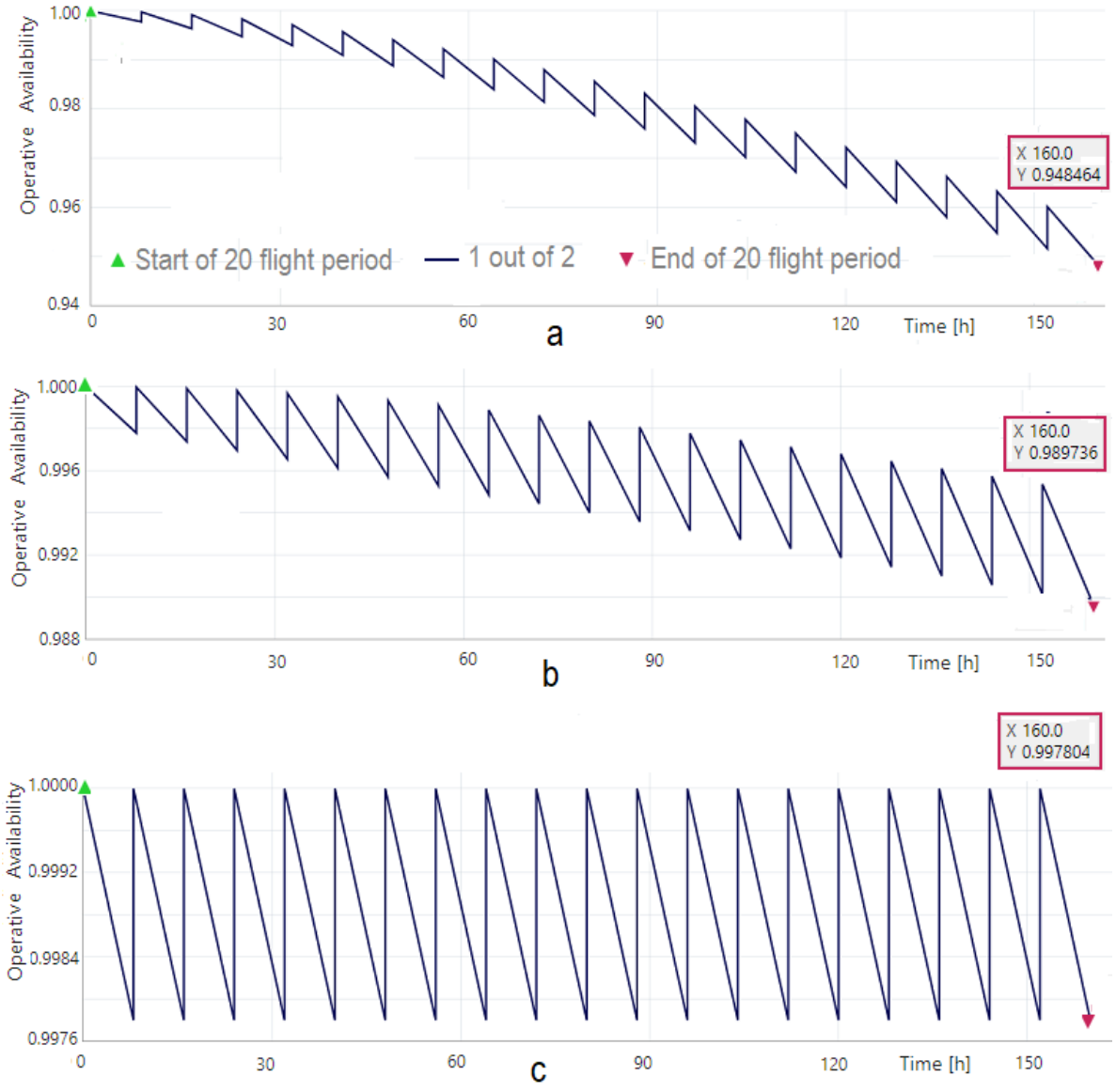

Fig. 4.17. Trend of Operative Availability ($Q_{step} = 20$) a). FDR=0.7 b). FDR=0.9 c). FDR=1.0.

.

5. CONCLUSION

In this paper, we presented software implemented analytic techniques of testability evaluation and joint testability and reliability analysis of the aircraft systems [31, 32]. The approach to testability evaluation relies on parametric analysis of FMEA data, including parameters as failure severity, fault detection method, failure type, impact on departure delay. We have presented two models for joint testability and reliability analysis. The first model is the fault-tree of BIT conformity considering functional properties and unreliability of built-in-test equipment. The second one is Markov reliability model considering imperfection of BIT fault coverage and cyclic operation mode specific for aviation systems. We have proposed a method for calculating operative availability taking into account the recovery scenario inherent in aviation. These models and methods are implemented in the web application of the Oracle database. The architecture of the database of system characteristics and FMEA reports was developed. Extended preliminary effort of developing database architecture was devoted to creation of meta specification of testability data as XML schema. The codes of programs defining and solving the models were implemented in the PL SQL language. The software pages with user interface components have smart grid layout adopted to varying screen resolutions both PC and mobile.

We have demonstrated the software capability on the sample system analysis including testability evaluation, reliability analysis of redundant assemblies with imperfect fault coverage, modeling of operative availability trend on inter-maintenance interval.

The software could be of practical value to designers of built-in-test systems and analysts of testability departments.

REFERENCES

1. GOST R 56081 (2014) *Izdeliya aviatsionnoy tekhniki. Bezopasnost' poleta, nadezhnost', kontroleprigodnost', ekspluatatsionnaya i remontnaya tekhnologichnost'. Poryadok normirovaniya i kontrolya pokazateley*. [State standard R 56081-2014. Aircraft items. Flight safety, reliability, testability and maintainability indices. Setting and control.] M.: Standartinform, [in Russian].
2. Beighelt, F., Franken, P. (1988) *Nadezhnost' i tekhnicheskoye obsluzhivaniye. Matematicheskiy podkhod.* [Reliability and maintenance. Mathematical approach.] M.:Radio i svyaz', [in Russian].
3. DoD 3235.1-H (1982) *Test and evaluation of system reliability, availability and maintainability*. Third edition. *OUSDRE/DDTE*, 3-3.
4. Amari, S.V., Myers, A.F., Rauzy, A., Trivedi, K.S. (2008) Imperfect Coverage Models: Status and Trends. In: Misra K.B. (eds) *Handbook of Performability Engineering*. Springer, London. https://doi.org/10.1007/978-1-84800-131-2_22
5. Li, X., Wan, H., Gong, Z., Wang, Z., Huang, H. (2011) Flight control system reliability study based on hIdden Markov Model imperfect fault coverage model-Hidden Markov Model. *Int.l Conf. on Quality, Reliability, Risk, Maintenance, and Safety Engineering*. Xi'an, 126–131, doi: 10.1109/ICQR2MSE.2011.5976582.
6. Xiong, X., Zhang, P. (2012) Reliability analysis of flight control system for large civil aircraft with Imperfect Fault Coverage Model. *Proc. IEEE Prognostics and System Health Management Conf. (PHM-2012 Beijing)*. Beijing, 1–5, doi: 10.1109/PHM.2012.6228935.
7. Lubkov, N.V., Spiridonov, I.B., Stepanyants, A.S. (2016) *Vliyaniye kharakteristik kontrolya na pokazateli nadezhnosti sistem.* [Influence of control characteristics on system reliability indicators.] // Trudy MAI, 85, [in Russian]. [Online]. Available: http://www.mai.ru/science/trudy/published.php?ID=67501.
8. Rani, P., Pahuja, G. (2018) Reliability Analysis of Flight Control System under Perfect and Imperfect Fault Coverage. *3rd IEEE Int. Conf. on Recent Trends in Electronics,*

Information & Communication Technology (RTEICT). 759-763.
9. A4A MSG-3. (2018) *Operator/Manufacturer Scheduled Maintenance Development*. Revision 2018.1.
10. Chekryzhev, N.V. (2015) *Osnovy tekhnicheskogo obsluzhivaniya vozdushnykh sudov: ucheb. posobiye*. [Aircraft Maintenance Fundamentals:study. manual]. Samara: Izd. SGAU, [in Russian].
11. MIL-STD-1629A (1984) *Procedures for performing a failure mode, effects and criticality analysis*. Notice 2.
12. ARP4761 (1996) *Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment*. SAE International.
13. Bidokhti, N., Loeser, M. (2012) Does your system have sufficient diagnostics coverage? *Proc. An. Reliability&Maintainability Symp.*. Reno, NV, 1–6, doi: 10.1109/RAMS.2012.6175510.
14. Liu, D., Zeng, Z., Huang, C., & Li, F. (2012) The testability modeling and model conversion technology based on multi-signal flow graph. *Proc. IEEE Prognostics & System Health Management Conf. (PHM-2012 Beijing)*. Beijing, 1–8, doi: 10.1109/PHM.2012.6228919.
15. Long, W., Yue, L., Yanling, Q. Tengfei, Q. & Minhao, W. (2017) A method of testability analysis and design based on FMEA extension. *13th IEEE Int. Conf. on Electronic Measurement & Instruments (ICEMI)*, Yangzhou, 361–367, doi: 10.1109/ICEMI.2017.8265968.
16. Viktorova, V.S., Stepanyants, A.S. (2010) *Proyektnyy analiz kontroleprigodnosti tekhnicheskikh sistem (teoriya, metody rascheta, programmnoye obespecheniye)*. [Design analysis of testability of technical systems (theory, calculation methods, software).] Sc.Pub. M.: IPU RAN, [in Russian].
17. Aviatsionnyy spravochnik AS 1.1. S1000DR-2013 (2013). *Mezhdunarodnaya spetsifikatsiya na tekhnicheskiye publikatsii, vypolnyayemyye na osnove obshchey bazy dannykh* [International Specification for Technical Publications Based on a Common Database]. M.: FGUP "NIISU" [in Russian].
18. International procedure specification for Logistic Support Analysis LSA (2014). *S3000L-B6865-03000-00.* Issue No 1.1. ASD, AIA.
19. International specification for technical publications using a common source database (2019). *S1000D-B6865-01000-00.* Issue No. 5.0. ASD, AIA.
20. W3School XML Tutorial. *XSD introduction*. [Online]. Available: https://www.w3schools.com/xml/schema_intro.asp
21. Viktorova, V.S., Spiridonov, I.B. (2015) *Universal'naya model' dannykh kontroleprigodnosti* [Generic Testability Data Model]. M.: IPU RAN, [in Russian].
22. S-TechEnterprises, LLC. (2005). *ATA 100 chapter and section headings*. [Online]. Available: http://www.s-techent.com/ATA100.htm
23. Molina, M., Carrasco, S., Martin, J. (2014). Agent-Based Modeling and Simulation for the Design of the Future European Air Traffic Management System: The Experience of CASSIOPEIA. *J.M. Corchado et al. (Eds.): PAAMS Workshops, CCIS 430*, 22–33 https://doi.org/10.1007/978-3-319-077673_3.
24. Henley, E.J., Kumamoto, H. (1996). *Probabilistic Risk Assessment and Management for Engineers and Scientists* (2nd ed.) NJ.: IEEEPress.
25. Viktorova, V.S., Stepanyants, A.S. (2016) *Modeli i metody rascheta nadezhnosti tekhnicheskikh sistem* [Models and methods of reliability analysis of technical systems]. M.: LENAND, [In Russian].
26. Ryabinin, I. (1976) *Reliability of engineering systems. Principles and Analysis*. English Translation. M.: Mir Publishers.
27. Ustinov, S.M. , Zimnitskiy, V.A. (2009) *Vychislitel'naya matematika* [Computational mathematics]. SPb.: BKHV-Peterburg, [In Russian].

28. Viktorova, V.S., Stepanyants, A.S. (2020) *Analiz nadezhnosti i effektivnosti mnogourovnevykh tekhnicheskikh sistem* [Analysis of reliability and efficiency of multilevel technical systems]. M.: LENAND [In Russian].
29. Stepanyants, A.S., Viktorova, V.S. (2019) Reliability analysis of systems with hybrid recovery and imperfect built-in-test. *Proc. of the 22nd Int. Conf. on Distributed Computer and Communication Networks: Control, Computation, Communications (DCCN-2019, Moscow)*. Cham: Springer, 413–423, [in Russian].
30. Petrukhin, B.P. (2014) *Osobennosti prognozirovaniya nadozhnosti sistem i ustroystv po metodike 217 Plus TM chast' 1. Modeli intensivnosti otkazov komponent* [Features of predicting the reliability of systems and devices according to the 217 Plus TM method. Part 1. Models of component failure rate]. *Datchiki i sistemy*, **6**, 37–43 [in Russian].
31. Viktorova, V.S., Lubkov, N.V., Stepanyants, A.S. (2017) Sowtware for testability analysis of aircraft functional systems: *Certificate of state registration of computer programs 2017660269 RF*; dated 20.09.2017.
32. Viktorova,V.S., Stepanyants, A.S. (2019) Analysis of the trend of operational availability of aircraft systems: *Certificate of state registration of computer programs 2019618035 RF*; dated 26.06.2019.

6. APPENDIX. TESTABILITY DATA XML SCHEMA DEFINITION

```
<?xml version="1.0" encoding="UTF-8"?>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:mstns="http://tempuri.org/TestabilityXMLSchema.xsd"
  xmlns="http://tempuri.org/TestabilityXMLSchema.xsd"
  elementFormDefault="qualified"
  targetNamespace="http://tempuri.org/TestabilityXMLSchema.xsd">
  <xs:annotation>
    <xs:appinfo>Testability Software</xs:appinfo>
    <xs:documentation xml:lang="en">
  This Schema defines a structure and properties of testability XML data
    </xs:documentation>
  </xs:annotation>
  <!- -Global Simple Types- ->
  <xs:simpleType name="TRIndex"> <!--Time parameters and reliability indices-->
    <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TRatio"> <!--Testability indices normalized from 0 to 1-->
    <xs:restriction base="xs:decimal">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="1"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TCodATA"> <!--Six or eight character code-->
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]{6}|[0-9]{6}-[0-9]{2}"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TLRUType"> <!--LRU type-->
    <xs:restriction base="xs:string">
      <xs:enumeration value="LRU"/> <!--line replaceable unit-->
```



```
      <xs:enumeration value="PL"/> <!--pipeline-->
      <xs:enumeration value="W"/> <!--wires-->
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TRSchema"> <!--Redundancy schema-->
    <xs:restriction base="xs:string">
      <xs:enumeration value="1out1"/> <!--single module-->
      <xs:enumeration value="1out2"/> <!--double module redundancy-->
      <xs:enumeration value="1out3"/> <!--triple module redundancy-->
      <xs:enumeration value="2out3"/> <!--majority redundancy-->
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TFDM"> <!--Failure detection methods-->
    <xs:restriction base="xs:string">
      <xs:enumeration value="BIT"/> <!--built-in-test-->
      <xs:enumeration value="Crew"/> <!--by flight crew-->
      <xs:enumeration value="Maint"/> <!--by repair team (Check A)-->
      <xs:enumeration value="None"/> <!--by repair team (Check B)-->
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TSeverity"> <!--Failure severity level-->
    <xs:restriction base="xs:string">
      <xs:enumeration value="Negligible"/> <!--severity level V-->
      <xs:enumeration value="Minor"/> <!--severity level IV-->
      <xs:enumeration value="Marginal"/> <!--severity level III-->
      <xs:enumeration value="Critical"/> <!--severity level II-->
      <xs:enumeration value="Catastrophic"/> <!--severity level I-->
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TPhases"> <!--Flight profile-->
    <xs:restriction base="xs:string">
      <xs:enumeration value="AllPhases"/>
      <xs:enumeration value="Takeoff"/>
      <xs:enumeration value="Climb"/>
      <xs:enumeration value="Cruise"/>
      <xs:enumeration value="Descent"/>
      <xs:enumeration value="Landing"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TFailureType"> <!--Type of LRU failure-->
    <xs:restriction base="xs:string">
      <xs:enumeration value="FF"/> <!--functional failure-->
      <xs:enumeration value="BIT_NOP"/> <!--no operation BIT failure-->
      <xs:enumeration value="BIT_FA"/> <!--false alarm BIT failure-->
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="TDelay"> <!-- Departure delay flag-->
    <xs:restriction base="xs:string">
      <xs:enumeration value="NO"/> <!--Failure does not impact on departure delay-->
      <xs:enumeration value="YES"/> <!--Failure impacts on departure delay-->
    </xs:restriction>
  </xs:simpleType>
  <!- -Global Complex Types- ->
```



```
<xs:complexType name="TFMDescriptor">
  <xs:sequence >
<!-- Failure mode descriptor (UID and names)-->
    <xs:element name="FID" type="xs:string"/> <!--failure mode identifier-->
    <xs:element name="FMName" type="xs:string"/> <!--failure mode name-->
<!-- Failure mode descriptor in centralized maintenance system (CMS)-->
    <xs:element name="FIDCMS" type="xs:string"/>
    <xs:element name="FMNameCMS" type="xs:string"/>
<!--Crew alerting system (CAS) message-->
    <xs:element name="CASmsg" type="xs:string" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
<!--Resolution list-->
<xs:complexType name="TLRUchain">
  <xs:sequence >
<xs:element name="CodATA" type="TCodATA" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence >
  </xs:complexType >
<!--Definition of FMEA Type-->
<xs:complexType name="TFMEA">
  <xs:all >
    <xs:element name="FMDescriptor" type="'TFMDescriptor"/>
    <xs:element name="FR" type="TRIndex"/> <!--failure rate-->
    <xs:element name="FDM" type="TFDM"/>
    <xs:element name="Severity" type="TSeverity"/>
    <xs:element name="Phase" type="TPhases"/>
    <xs:element name="FailureType" type="TFailureType" />
    <xs:element name="Delay" type="TDelay" />
    <xs:element name="Resolution" type="xs:positiveInteger" />
    <xs:element name="LRUchain" type="TLRUchain" />
  </xs:all>
</xs:complexType>
<!--Definition of LRU Type-->
<xs:complexType name="TLRU">
  <xs:all >
    <xs:element name="LRUName" type="TDescriptor"/>
    <xs:element name="PartNumber" type="xs:string"/>
    <xs:element name="SerialNumber" type="xs:string"/>
<!--UID in Centralized Maintenance System-->
    <xs:element name="IDCMS" type="xs:string"/>
    <xs:element name="LRUType" type="TLRUType" />
    <xs:element name="RSchema" type="TRSchema" />
<!--Mean Time to First Failure by Specification-->
    <xs:element name="MTTFF" type="TRIndex" />
<!--Mean Time to Repair by Specification-->
    <xs:element name="MTTR" type="TRIndex" />
    <xs:element name="FMEAs" type="TFMEAs" />
  </xs:all>
</xs:complexType>
<!--Definition of System Type-->
<xs:complexType name="TSystem">
  <xs:all >
    <xs:element name="SName" type="TDescriptor"/>
```



```
    <!--Mean Time to First Failure by Specification-->
        <xs:element name="MTTFF" type="TRIndex"/>
        <xs:element name="FDR" type="TRatio"/> <!--Fault Detection Ratio-->
        <xs:element name="FIR" type="TRatio"/> <!--Fault Isolation Ratio-->
        <xs:element name="BRF" type="TRatio"/> <!--BIT Reliability Factor-->
        <xs:element name="FAF" type="TRatio"/> <!--False Alarm Factor-->
        <xs:element name="LRUs" type="TLRUs" />
      </xs:all>
    </xs:complexType>
    <!--Definition of Project Type-->
    <xs:complexType name="TProject">
      <xs:all >
        <xs:element name="PrName" type="TDescriptor"/>
        <xs:element name="Tf" type="TRIndex"/> <!--flight duration-->
        <xs:element name="Tc" type="TRIndex"/> <!--A check interval-->
        <xs:element name="Tb" type="TRIndex"/> <!--B check interval-->
        <xs:element name="Tp" type="TRIndex"/> <!--C check interval-->
        <xs:element name="Tr" type="TRIndex"/> <!--D check interval-->
        <xs:element name="Systems" type="TSystems" />
      </xs:all>
    </xs:complexType>
    <!--Set of FMEAs of the LRU-->
    <xs:complexType name="TFMEAs">
      <xs:sequence >
< xs:element name="FMEA" type="TFMEA" minOccurs="1" maxOccurs="unbounded" />
      </xs:sequence >
      </xs:complexType >
    <!--Set of LRUs of the System-->
    <xs:complexType name="TLRUs">
      <xs:sequence >
< xs:element name="LRU" type="LRU" minOccurs="1" maxOccurs="unbounded"  />
      </xs:sequence >
      </xs:complexType >
    <!--Set of Systems of the Project-->
    <xs:complexType name="TSystems">
      <xs:sequence >
< xs:element name="System" type="TSystem" minOccurs="1" maxOccurs="unbounded" />
      </xs:sequence >
      </xs:complexType >
    <!--Definition of the set of projects-->
    <xs:complexType name="TProjects">
      <xs:sequence >
< xs:element name="Project" type="TProject" minOccurs="1" maxOccurs="unbounded" />
      </xs:sequence >
      </xs:complexType >
    <!--Global Element/Root of XML document-->
      <xs:element name="Projects" type="TProjects" />
    </xs:schema>
```

