

ISO/IEC JTC1/SC7
Software Engineering
Secretariat: CANADA (SCC)

ISO/IEC JTC1/SC7 N1444
September 22, 1995

TITLE: **WD and CD Registration Ballot**
Measurement and rating of performance of
computer-based software systems

SOURCE: **SC7/WG6**

WORK ITEM: **07.36**

STATUS: **WD and CD registration ballot**

REFERENCE: **JTC1 N3371**

ACTION: **Letter Ballot must be returned to the SC7 Secretariat no later than**
1996-01-05. Discussions on this document will be held during the
meeting of SC7/WG6 on November 27 - December 1 1995 in Dublin,
Ireland. National Body comments will thus be appreciated.

ISO/IEC JTC1/SC7
Software Engineering
Secretariat: CANADA (SCC)



1995-09-22

LETTER BALLOT

From the member body of:

Proposal:

That the attached WD document SC7 N1444 titled:
"Measurement and rating of performance of computer-
based software systems" be registered as a CD.

*We support the proposal as presented

OR

*We support the proposal with the attached comments

OR

*We do not support the proposal for the technical reasons
attached to this ballot

OR

*We abstain from voting
(*P* members have an obligation to vote)

DUE DATE: 1996-01-05

Address reply to: ISO/IEC JTC1/SC7 Secretariat
F. Coallier
Bell Canada - Acquisitions
2265 Roland Therrien, Room 226, Longueuil (Quebec) Canada J4N 1C5
Tel.: +1 (514) 448-5100 Fax: +1 (514) 448-2090 or +1 (514) 647-3163
fcoallie@qc.bell.ca

ISO/IEC JTC1/SC7
Software Engineering
Secretariat: CANADA (SCC)

***DELETE WHICHEVER DOES NOT APPLY**

Address reply to: ISO/IEC JTC1/SC7 Secretariat
F. Coallier
Bell Canada - Acquisitions
2265 Roland Therrien, Room 226, Longueuil (Quebec) Canada J4N 1C5
Tel.: +1 (514) 448-5100 Fax: +1 (514) 448-2090 or +1 (514) 647-3163
fcoallie@qc.bell.ca

ISO/IEC JTC1 / SC7 / WG6

Evaluations and Metrics

6/N - 328

NP 14756 / Project 7-36

TITLE:	„Measurement and rating of performance of computer-based software systems“
Part 1	„Principles“
Part 2	„Detailed description of the measurement and rating“

DATE: August, 1st, 1995

VERSION: 4.1

SOURCE: JTC1 / SC7 / WG6

Prof. Dr. Werner. Dirlewanger (Editor)
Kassel University / Dep. of Math. & Computer Science
P.O.Box 10 13 80
D 34109 Kassel
- Germany -
Tel: +49 561 804 4349 / Fax: +49 561 804 4302

Reinhold Schluempmann (Editor)
Deutsche Telekom AG / IZ Darmstadt
P.O.Box 10 14 54
D 64214 Darmstadt
- Germany -
Tel: +49 6151 818 4323 / Fax: +49 6151 818 4332
Email: schluepmann@04.dmst03.telekom400.dbp.de

REFERENCE: 6/N-245, 6/N-255, 6/N-257, 6/N-277, 6/N-292,
JTC1-N 3371, 6/N-310

STATUS: Working Draft

The Document was revised based on the comments,
given before and at the Brisbane meeting (June 1995).

ACTION: To be reviewed

at SC7 / WG6 meeting in Dublin/Ireland (November 1995)

ISO/IEC JTC1 / SC7 / WG6

Evaluations and Metrics

NP 14756 / Project 7-36

Measurement and rating
of
performance of computer-based software systems

Part 1: Principles

(pages 1 to 18)

Contents

Foreword

Introduction

- 1 Scope
- 2 Normative references
- 3 Principles for measurement and rating
 - 3.1 The measurement
 - 3.1.1 Test environment
 - 3.1.1.1 Standard configuration
 - 3.1.1.2 Detailed hardware specification
 - 3.1.1.3 Complete specification of basic data
 - 3.1.2 User emulation
 - 3.1.2.1 Random behaviour of the users
 - 3.1.2.2 Parameter set of the user entity
 - 3.1.2.3 Remote terminal emulator
 - 3.1.3 The measurement procedure
 - 3.1.3.1 Three phases of the measurement procedure: Warm up, rating interval, supplementary run
 - 3.1.3.2 Writing a logfile
 - 3.1.4 Proof of validity of the measurement
 - 3.1.4.1 Proof of the remote terminal emulator's accuracy
 - 3.1.4.2 Proof of the measurement results accuracy
 - 3.1.4.3 Proof of the CBSS's work correctness
 - 3.2 Basic data for rating
 - 3.2.1 Execution time requirement of the users
 - 3.2.2 The reference configuration for rating software efficiency
 - 3.2.2.1 Reference configuration for assessing application software
 - 3.2.2.2 Reference configuration for assessing system software
 - 3.3 Estimation of the performance values of a CBSS
 - 3.3.1 Throughput
 - 3.3.2 Mean execution time
 - 3.3.3 Timely throughput
 - 3.4 Rating of the estimated performance values of the CBSS
 - 3.4.1 Computation of the performance reference values
 - 3.4.1.1 Mean execution time reference values
 - 3.4.1.2 Throughput reference values
 - 3.4.2 Computation of the performance rating values
 - 3.4.2.1 The throughput rating values
 - 3.4.2.2 The mean execution time rating values
 - 3.4.2.3 The timeliness rating values
 - 3.4.3 Rating of the performance of the CBSS
 - 3.4.4 Directions
 - 3.4.4.1 Measurement of all performance values
 - 3.4.4.2 Proper definition of the reference configuration

Foreword

< Remark of the editor: Foreword is written by ISO Central Secretariat. >

Introduction

In both the planning and using of data processing systems, the speed of execution is a very interesting and significant property. This property is influenced greatly by the efficiency of the software used in the system. Measuring the speed of the system as well as the influence of the efficiency of the software is of elementary interest.

To measure the influence which software has on the time behaviour of a data processing system it is necessary to measure the time behaviour of the whole system. Based on the metrics of the measurement procedure proposed in this standard it is possible to define and to compute the values of the time efficiency of the software.

It is important to estimate the time behaviour characteristics in a reproducible way. Therefore it is not possible to use human users in the experiment. One reason is that human users cannot reproduce longer phases of computer usage several times without deviations of characteristics of usage. Another reason is that it would be too expensive to carry out such experiments with human users if the job or task stream comes from many users. Therefore an emulator is used which emulates all of the users by use of a second data processing system.

This means that measurement and rating of performance according to this standard needs a tool. This tool is the emulator which has to be constructed according to the specifications in this standard. It has to be proven that the emulator used actually fulfils these specifications.

All of the relevant details of this experiment are recorded in a logfile by the user emulator. From this logfile then the values which describe the time behaviour (for instance response times and throughput values) can be computed. From these performance values the interesting software efficiency rating values will be computed.

This multipart standard consists of two parts:

- Part 1: Principles
- Part 2: Detailed description of the measuring and rating method

Part 1 describes the principles for measurement and rating used in this standard, the used metrics and the defined performance terms and rating terms.

Part 2 specifies the measurement and rating procedure in detail. In addition it is a guide to everyone who plans to carry out a measurement and rating procedure. It is also a guide to developing a workload according to the specifications of this standard. This may be of interest if there is no proper workload available or if an already existing workload should not be used for some reason.

1 Scope

The subjects of this standard are computer-based software systems (shortened CBSS). A CBSS is a data processing system as it is seen by its users, i. e. the users at the terminals. A CBSS includes the hardware and all of the software (system software and application software) which is needed to realize the data processing functions needed by the users.

This standard specifies user oriented performance terms, i. e. terms which allow the execution speed of user orders (tasks) to be described. It focuses on

- application software
- system software
- turn-key systems
(i. e. systems consisting of an application software and the hardware for which it was designed)
- general data processing systems.

A further aim of this standard is to specify a method to measure and to rate such user oriented performance values. The rating can be done with respect to the users requirements or by comparing two or more measured systems (types or versions).

Following the rules of this standard each measurement of a data processing system under test is done within a reproducible steady state. Therefore additional performance values, i. e. utilisation values, mean instruction rates, path lengths, cache hitrates, queuing times and service times, may be estimated - if someone is interested in them - in addition to the useroriented performance values as they are defined in this standard. Since the interest of this standard focuses on the time behaviour of CBSS no proposals on how to measure such internal performance values will be made in parts 1 and 2. But additional parts which deal with the internal performance values are planned to be attached later on.

This standard describes the rules

- how to design an useroriented benchmark test using a transaction oriented, batch oriented or mixed workload
- how to define such a workload,
- how to perform the measurement procedure,
- how to rate the measured results.

Workloads fulfilling the specifications of this standard and having a sufficient general structure may be used as standard workloads. Such standard workloads are planned to be defined later on and to be added as separate parts to this multipart standard. They may be used to measure and to rate performance of CBSS used in specific fields. E. g. a standard workload for word-processing may be used to compare the time efficiency of different software products or different versions of the same product running on the same hardware system. But such a standard workload may also be used if always applying the same application software version and the same hardware to compare the efficiency of the system software.

Affected interests are mainly those of

- evaluators and
- developers

of CBSSs (they apply to all parts of this standard)

but also

- buyers (i. e. users)
- system integrators

of CBSS (they primarily need part 1 for a quick reference).

2 Normative references

- ISO 8402
- IEEE 802
- ISO 19121

3 Principles of measurement and rating

3.1 The measurement

3.1.1 Test Environment

3.1.1.1 Standard configuration

The CBSS to be measured consists of the hardware, the system software and the application software, all of them in a standard configuration. None of the components may have any changes or special modifications with respect to getting better results in the measurement.

3.1.1.2 Detailed hardware specification

All of the hardware components (including wide area and local area networks) and all of the software components shall be specified in full detail.

3.1.1.3 Complete specification of basic data

All of the needed basic data (i. e. data files, content of a used data base system,...) shall be described and stored in full detail.

3.1.2 User emulation

3.1.2.1 Random behaviour of the users

In reality the users do not work strictly deterministic. This fact has a great influence on the system under test and results in internal queuing problems.

In view of the CBSS the users have a random behaviour (i. e. randomly changing think times, randomly changing sequences of tasks and so on). In accordance with this fact, the users behaviour shall not be described by fixed sequences of jobs, tasks, transactions or process starts, each having a fixed think time before its initialisation and similar properties. The user's behaviour shall be described by their statistically significant properties. This yields descriptive values as for instance:

- think time mean value and its standard deviation,
- relative frequencies of the job or task types.

3.1.2.2 Parameter set of the user entity

The user entity to be emulated for the measurement shall be described by a set of parameters:

- number of user types,
- number of users of each type,
- job or task types submitted by the users,
- relative frequencies of submission of the job or task types by each of the user types,
- think time mean values and standard deviations of all of the job or task types for each of the user types,
- the required accuracy of the emulators work
- the required statistic significance of the measurement results (among others: required confidence intervals d)
- and further details (see part 2 of this standard).

The complete set of these values is the so called workload parameter set. It is described in detail in part 2 of this standard.

3.1.2.3 Remote terminal emulator

The user entity (see figure 1) shall be emulated by use of a remote terminal emulator (RTE). This RTE is connected to the CBSS under test (system under test, shortened SUT) via the real local area network or wide area network and the real terminal lines. The RTE shall be implemented by use of a separate data processing system. It shall not be implemented by a program running on the SUT itself.

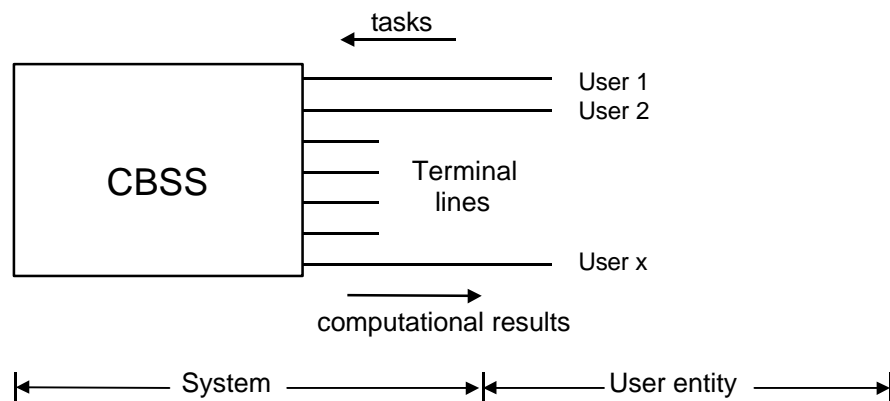


Fig. 1 CBSS in real operation

The RTE is driven by the set of workload parameters described in subclause 3.1.2.2. The RTE emulates each of the users separately and in a real time mode (see figure 2). The job or task sequences are random, but realizing the relative job or task frequencies as defined in the workload parameter set. For special cases this job or task stream may contain fixed sequences (so called task chains) depending on the necessities of the function of the application software. The think times shall be varied randomly but realizing the mean values and standard deviations as defined in the workload parameter set. (Details are described in part 2 of this standard.)

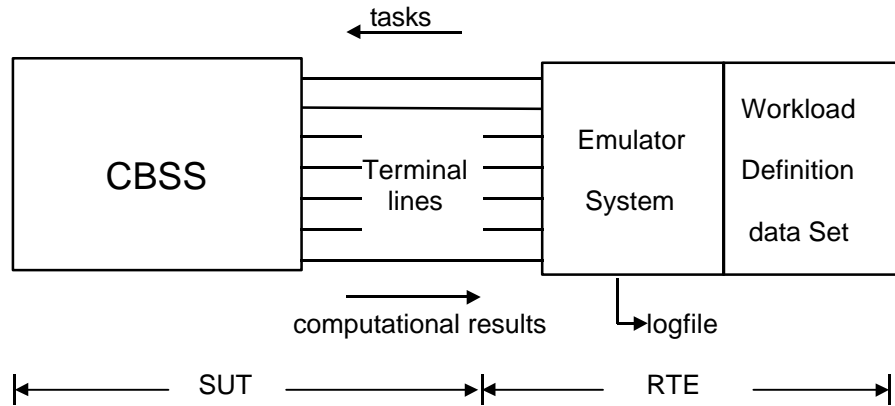


Fig. 2 CBSS in measurement

3.1.3 The measurement procedure

3.1.3.1 Three phases of the measurement procedure: Warm up, rating interval, supplementary run

The measurement begins with a warm up phase. When the system has come to the so called steady state then the rating interval starts. Its duration has to be chosen appropriately to the application which is represented by the software under test. At the end of the rating interval the RTE shall not be stopped yet. It has to continue the job or task initiation as specified by the workload parameters. It shall only be stopped when all of those jobs or tasks which had been initiated within the rating interval are completed. The time from the end of the rating interval to the earliest moment when the emulator is allowed to be stopped is called supplementary run (see figure 3).

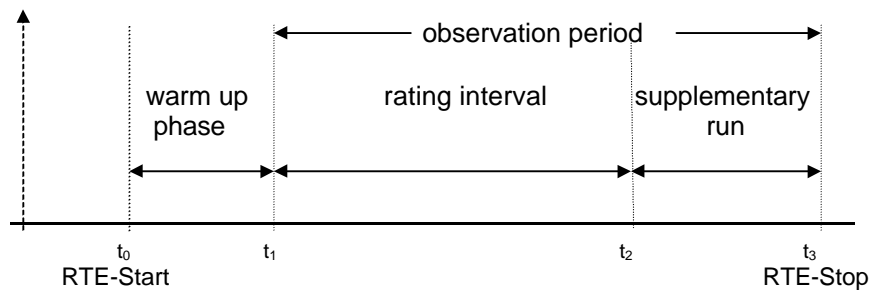


Fig. 3 Three phases of the measurement procedure

3.1.3.2 Writing a logfile

The needed history data of all of the jobs or tasks which were started within the time of beginning the rating interval to the end of the supplementary run shall be recorded in the log file. The history data are: Start time of each job or task, its type, the identification of the user who started it, the time of completion, the job's computation results and so on. For details see part 2 of this standard.

3.1.4 Proof of validity of the measurement

As stated above, the method of this standard makes no effort to get measurement runs which are reproducible in the microscopic details because this does not correspond to the situation in the reality of data processing. But with respect to the macroscopic (i. e. the statistic) properties the method is very correct. This is ensured by use of two statistical tests (as described in the subclauses 3.1.4.1 and 3.1.4.2) and the proof of the CBSS's correct work (see subclause 3.1.4.3).

3.1.4.1 Proof of the remote terminal emulator's accuracy

The actual values of mean think times, relative job or task frequencies and think time standard deviations shall be computed from the logfile. These actual values shall be compared to the values defined in the workload parameter set (see subclause 3.1.2.2). The differences of actual values to defined values shall be computed. These differences may not exceed a defined limit DELTA (as an example: DELTA = 1%). Otherwise the measurement is invalid due to the insufficient precision of the remote terminal emulator. In the latter case the measurement has to be repeated using a better working RTE.

3.1.4.2 Proof of the measurement result's accuracy

Whenever a data processing system is working the details of time behaviour are random. This is due to changing disc spindle resolutions, little differences of the clock frequency, random data collisions on buses and networks, random use of cache stores and so on. Therefore the actual response times or execution times of jobs or tasks differ randomly to a certain extent. This means that in principle a measurement run can't deliver the exact values.

Every run delivers some statistical errors in the estimated values (i. e. in the response time mean values). But the theory of mathematical statistics presents a tool called confidence interval to describe the amount of the possibly involved error, with respect to a defined confidence coefficient. (Details see part 2 of the standard.) The confidence intervals are computed from the logfile. The computed values shall be compared with the values "d" as defined in the workload parameter set. The "d"-values are the maximum accepted statistic errors (as an example $d = 10\%$).

If the computed values exceed the defined d-values the measurement results are not sufficiently significant. The typical situation in this case is that the measurement interval chosen was too short. A new measurement run has to be done, using a longer measurement interval resulting in more samples and a greater chance of getting a statistically significant result.

3.1.4.3 Proof of the CBSS's work correctness:

For all of the jobs or tasks which were initiated within the measurement interval and in the supplementary run (see subclause 3.1.3.1) it has to be proven that all of these jobs or tasks have been done completely and have produced correct computational results. This proof shall be done by use of the recorded data in the logfile and the stored computing result data in the SUT.

3.2 Basic data for rating

3.2.1 Execution time requirements of the users

No application of a computer is useful in practice if it works too slow in view of the user's requirements. In order to rate a CBSS with respect to its efficiency it is necessary to describe the user requirements. This is done by use of so called timeliness functions.

Remark:

An example of a timeliness function is:

The completion of online transactions of a defined type has to be done within 1.5 seconds in at least 90% of the transactions; up to 10% of the response times may have more than 1.5 seconds but not more than 4.0 seconds; response times greater than 4.0 seconds are not accepted. (For the general definition of the timeliness function: see part 2 of this standard.)

The timeliness requirements for each of the job or task types (interactive tasks as well as batch jobs) shall be defined for each of the users by use of timeliness functions.

3.2.2 The reference configuration for rating software efficiency

If in addition to the performance of a CBSS it is also planned to assess the efficiency of some parts of the software then the so called reference configuration shall be defined.

3.2.2.1 Reference configuration for assessing application software efficiency

For assessing application software efficiency the definition of the reference configuration shall be done as follows:

The design of a software generally is done with respect to a computer of defined type and size which is assumed to be available to the user. As an example: Constructing a bookkeeping program the designer has to consider if it is intended to work on a little PC or on a large PC, on a PC network including a data server, on a midrange multi-user system, on a mainframe computer and so on. The computer planned to be in use greatly influences the software design. I. e.: An application software package always is designed to work on a particular hardware configuration and its system software.

Therefore an essential part of the basic information to rate the efficiency of application software is the (reference-) hardware on which the software is intended to be used and the (reference-) system software (i. e. operating system, compiler type, network system,...). These components have to be defined and completely listed: This is the reference configuration (i.e. hardware and system software) for the rating of application software system.

The results of the estimation of application software efficiency always refer to the used reference configuration.

3.2.2.2 Reference configuration for assessing system software efficiency

For assessing system software efficiency a reference configuration shall be defined. The components of it shall be defined in detail and comprehensively. The procedure is analogous to that of subclause 3.2.2.1: The reference configuration consists of the (reference-) hardware configuration and the (reference-) application software.

The results of the estimation of system software efficiency always refer to the used reference configuration.

3.3 Estimation of the performance values of a CBSS

The following performance values shall be computed from the recorded logfile (logfile see subclause 3.1.3.2). To estimate these performance values all those tasks shall be taken into account the start of which is within the rating interval (rating interval see subclause 3.1.3.1).

3.3.1 Throughput

For each of the job or task types the average number of jobs or tasks per time unit shall be computed with respect to the rating interval. In case of m job or task types this yields the m values

$$B(1), B(2), \dots, B(m) .$$

$B(j)$ is the throughput of the j-th job or tasks type.

3.3.2 Mean execution time

For each of the job or task types the mean execution time ("average") shall be computed with respect to the rating interval. In case of m job or task types this yields the m values

$$T_{ME} (1), T_{ME} (2), \dots, T_{ME} (m) .$$

$T_{ME} (j)$ is the mean execution time of the j-th job or task type.

In case of a batch job the execution time is the elapsed time for the completion of the job. In case of an interactive task the execution time is the response time of the CBSS. For details of the definition of "execution time" see part 2 of this standard.

3.3.3 Timely throughput

The timely throughput is the average number of all of the jobs or tasks per time unit which have been completed timely with respect to the defined timeliness function of this job or task type and with respect to the rating interval. In case of m job or task types this yields the m values

$$E(1), E(2), \dots, E(m) .$$

For details on how to compute the E-values see part 2 of this standard.

3.4 Rating of the estimated performance values of the CBSS

3.4.1 Computation of the performance reference values

3.4.1.1 Mean execution time reference values

The reference values for the execution times result from the user's time requirements. If - as shown in the example of subclause 3.2.1 - 90% of the tasks are allowed to have an execution time of 1.5 seconds and the rest are allowed to have a execution time of 4.0 seconds then the reference value for the mean execution time is

$$0,90 * 1.5 + 0.1 * 4.0 = 1.75 \text{ seconds} .$$

In an analogous manner the reference values shall be computed for all of the job or task types. In case of m types this yields the m execution time reference values

$$T_{REF} (1), T_{REF} (2), \dots , T_{REF} (m) .$$

3.4.1.2 Throughput reference values

The reference values for the throughput result from the hypothetical situation that the mean execution times of the CBSS (for all of the job or task types) exactly equals the execution time reference values (see subclause 3.4.1.1). In case of m task types this yields the m throughput reference values

$$B_{REF} (1), B_{REF} (2), \dots , B_{REF} (m) .$$

The formula to compute this reference values from the workload parameter set (see subclause 3.1.2.2) is described in part 2 of the standard. The throughput reference values shall be computed for all of the job or task types.

3.4.2 Computation of the performance rating values

3.4.2.1 The throughput rating values

The quotient of the measured throughput to the reference throughput delivers the throughput rating value for each of the job or task types:

$$R_{TH}(\dots) = B(\dots) / B_{REF}(\dots) .$$

In case of m job or task types this yields m throughput rating values

$$R_{TH}(1), R_{TH}(2), \dots , R_{TH}(m) .$$

3.4.2.2 The mean execution time rating values

The quotient of the mean execution time reference value to the measured mean execution time value delivers the mean execution time rating value for each of the job or task types:

$$R_{ME}(\dots) = T_{REF}(\dots) / T_{ME}(\dots) \quad .$$

In case of m job or task types this yields m mean execution time rating values:

$$R_{ME}(1), R_{ME}(2), \dots, R_{ME}(m) \quad .$$

3.4.2.3 The timeliness rating values

The quotient of the value of timely throughput of a job or task type to the measured (total) throughput of this job or task type delivers the timeliness rating value for each of the job or task types:

$$R_{TI}(\dots) = E(\dots) / B(\dots) \quad .$$

In case of m job or task types this yields m timeliness rating values:

$$R_{TI}(1), R_{TI}(2), \dots, R_{TI}(m) \quad .$$

3.4.3 Rating of the performance of the CBSS

The rating values $R_{TH}(\dots)$, $R_{ME}(\dots)$, $R_{TI}(\dots)$ shall be computed for each of the job or task types. This yields $3 * m$ values.

Whenever one of the rating values $R_{TH}(\dots)$ is below 1 then the throughput of the related job or task types is too weak.

Whenever one of the rating values $R_{ME}(\dots)$ is below 1 then the mean execution time of the related job or task types is too long .

Whenever one of the rating values $R_{TI}(\dots)$ is below 1 there are more execution times which are too long than is acceptable with respect to the user's requirements (which are defined by the timeliness functions, see subclause 3.2.1). Remark: This may happen even in the situation that all of the throughput rating values and all of the mean execution time rating values for the related job or task type are sufficient.

3.4.4 Directions

3.4.4.1 Measurement of all performance values

In general the rating values for throughput, mean execution time and for timeliness are not or only slightly correlated. Therefore they can not be computed from one another. All of these values shall be separately estimated by a measurement in the manner, explained above previously.

3.4.4.2 Proper definition of the reference configuration

It may happen, that one or more rating values stay below 1 even if many efforts are made to improve the efficiency of the tested software. In this situation the internal speed of the hardware of the chosen reference configuration ("reference configuration" see subclause 3.2.2) is too weak. To improve the situation a faster computer has to be used as a reference. That means that a new "reference configuration" has to be defined.

3.4.5 Assessing the time behaviour of the software

For assessing software efficiency of the actual CBSS the estimated rating values $R_{TH}(\dots)$, $R_{ME}(\dots)$, $R_{TI}(\dots)$ have to be compared to those of the reference configuration (see subclauses 3.2.2.1 and 3.2.2.2). The steps for an assessment are shown in figure 4.

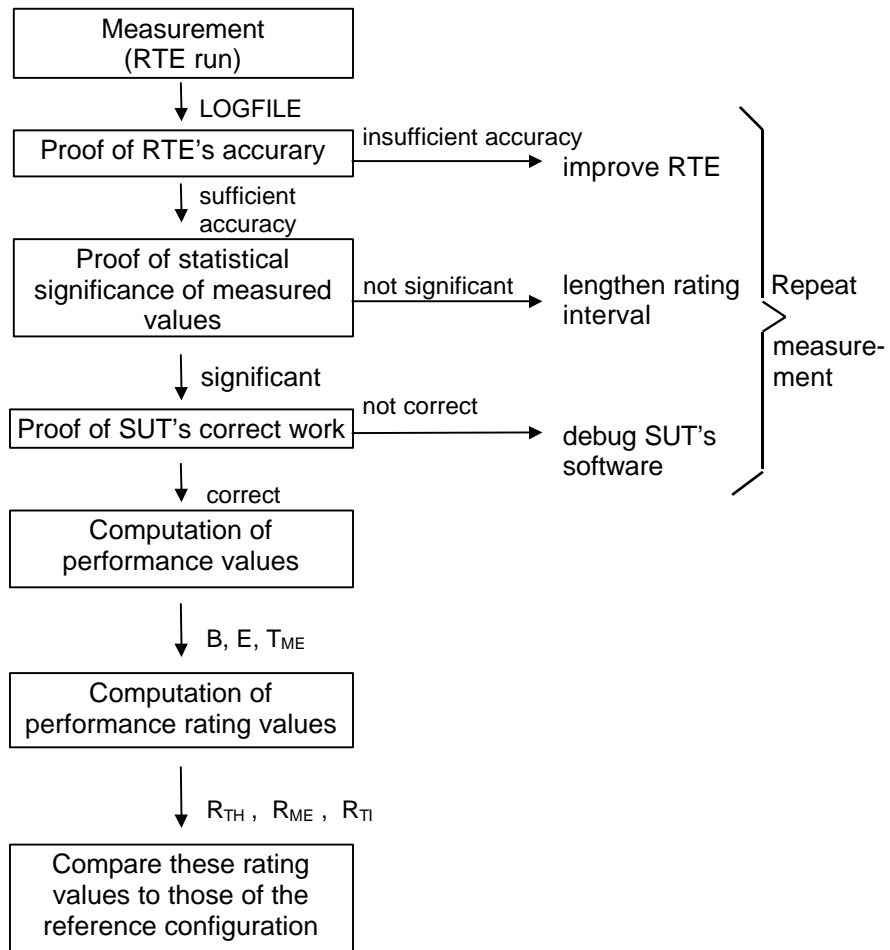


Figure 4 Assessing of software efficiency with respect to its time behaviour

ISO/IEC JTC1 / SC7 / WG6

Evaluations and Metrics

NP 14756 / Project 7-36

Measurement and rating
of
performance of computer-based software systems

Part 2:
Detailed description of the
measurement and rating procedure

(pages 1 to 52)

Contents

Introduction

- 1** Scope
- 2** Normative References
- 3** Definitions
 - 3.1 Computer-based software system (CBSS)
 - 3.2 User
 - 3.3 Workload (WL) and workload parameter set (WPS)
 - 3.4 System under test (SUT)
 - 3.5 Remote terminal emulator (RTE)
 - 3.6 User type (UT)
 - 3.7 Task
 - 3.8 Task type
 - 3.9 Task chain
 - 3.10 Chain type
 - 3.11 Execution time
 - 3.12 Think time
 - 3.13 Job mode (JM)
 - 3.14 Relative chain type frequency
 - 3.15 Timeliness function (TF)
 - 3.16 Mean execution time
 - 3.17 Throughput
 - 3.18 Timely throughput
 - 3.19 Throughput reference value
 - 3.20 Mean execution time reference value
 - 3.21 Throughput rating value for the j-th task type
 - 3.22 Mean execution time rating value for the j-th task type
 - 3.23 Timeliness rating value for the j-th task type
- 4** Abbreviations and symbols
 - 4.1 Abbreviations
 - 4.2 Symbols
- 5** Input needed for Measurement
 - 5.1 Workload Description
 - 5.1.1 The workload parameter set (WPS)
 - 5.1.1.1 Basic parameters
 - 5.1.1.2 Definitions of the task types
 - 5.1.1.3 Definitions of the task chain types
 - 5.1.1.4 Relative chain frequencies
 - 5.1.1.5 Mean think times and their standard deviations
 - 5.1.2 The (application) programs
 - 5.1.3 The list of additionally needed software for the measurement run
 - 5.1.4 The stored data

- 5.2 Description of the system under test (SUT)
 - 5.2.1 Hardware architecture and configuration
 - 5.2.2 System software configuration
 - 5.2.3 Application software configuration
- 5.3 The correct output of the computations in the measurement run
- 6** The Measurement Procedure
 - 6.1 Steps of the measurement procedure
 - 6.2 Modification of the rating interval definition
- 7** Output of the Measurement Process
 - 7.1 Measurement logfile
 - 7.2 Computation result file
- 8** Estimation of Performance values
 - 8.1 Throughput
 - 8.2 Mean execution time
 - 8.3 Timely throughput
- 9** Rating of the measured performance values of the SUT
 - 9.1 Computation of the performance reference values
 - 9.1.1 Mean execution time reference values
 - 9.1.2 Throughput reference values
 - 9.2 Computation of the rating values
 - 9.2.1 Computation of the throughput rating values
 - 9.2.2 Computation of the execution time rating values
 - 9.2.3 Computation of the timeliness rating values
 - 9.3 Rating
 - 9.3.1 Rating in detail
 - 9.3.1.1 Throughput rating
 - 9.3.1.2 Mean execution time rating
 - 9.3.1.3 Timeliness rating
 - 9.3.2 Overall rating
- 10** Validation of Measurements
 - 10.1 Validation of the accuracy of the RTE
 - 10.2 Validation of the computation correctness of the SUT
 - 10.3 Validation of the statistical significance of the measurement results

Annexes

- Annex A (normative) Specification of the RTE's basic functions
- Annex B (normative) Additional formula/methods of computation
- Annex C (normative) Standardized format of the workload description
- Annex D (normative) Standardized format of the logfile
- Annex E (informative) Programs
- Annex F (informative) Examples of workloads

Introduction

The principles for estimating objective and exact performance values of computer-based software systems and rating values for their performance were set up in part 1 of this international standard. In order to be able to estimate unique and doubtless values for the performance terms and rating terms it is necessary to define all details of the steps of the measurement experiment and of the rating procedure.

And - in addition - it is necessary to describe a proper set of functional details for the remote terminal emulator which is the tool carrying out the measurement.

Finally - for reasons of clarity and exactness - it is necessary to define some terms which are also commonly used in a more precise way. It is unavoidable to make such definitions. But it is clear that they are only valid locally within this standard.

All of these descriptions and definitions will be done in this part 2 of the standard.

1 Scope

This part 2 of the standard is the reference for those who intend to carry out measurements based on this standard. It is also the reference for those who want to implement a RTE or prove a RTE according to this standard. These persons are primarily evaluators and developers of computer-based software systems (CBSS). It also is the reference for other persons who are interested in the performance of CBSSs, for instance buyers or system integrators.

This part 2 of the standard defines - as far as it has not yet been done in part 1 of the standard - the terms needed to carry out the measurement using any type of workload (i. e. transaction-, demand-, dialog-, process-, batch-oriented or others). In addition - as far as it has not yet been done in part 1 of the standard - the terms which are needed to rate the measured performance values with respect to the defined user requirements are defined.

Not all of these values are always needed for carrying out a measurement and rating procedure. For instance if a simple workload having only a few interactive task types or only a simple sequence of batch jobs is used, then only a small subset of all of the terms and values which are defined is needed. But this method also allows the measuring and rating of a large and complex CBSS processing a complex job or task stream which is generated by a large set of many different users. As far as it is necessary the definitions are completed by mathematical terms. This is in order to obtain an exact mathematical basis for the computations of performance and rating values and for checking the correctness of the measurement run and rating steps as well as for the (statistical) significance of the estimated performance values and rating results.

This part 2 also specifies in detail the rules on how to prepare and carry out the measurement procedure. Along with a description on how to analyse the measured values the formulas for computing the performance values and the rating values will also be provided.

This part 2 of the standard also defines the rules for user emulation according to which the remote terminal emulator (RTE) shall work.

The result of a measurement consists of the estimated performance values. These are throughput values and execution time values. The final result of performance assessment of a CBSS consists of the rating values. They are gained by comparing the estimated performance values with the user's requirements. In addition it is possible - if desired - to rate the performance values of the CBSS under test by comparing them with those of a reference CBSS (for instance having the same hardware configuration but another version of the application program with the same functionality).

The result of the rating procedure is a set of values, each of them higher, lower or equal to 1. The rating values have the meaning of "better than", "worse than" or "equal to" the defined requirements (or the properties of a second CBSS used as a reference). The final estimated set of rating values assesses each of the job or task types which are defined in the workload separately.

2 Normative References

- ISO 8402
- IEEE 802
- ISO 19121

3 Definitions

Term (abbreviation in brackets)	Definition
3.1 Computer-based software system (CBSS)	<p>This is a software system running on a computer. I. e. it is a data processing system as it is seen by its users at the terminals (or at equivalent machine-user-interfaces). A CBSS includes the hardware and all of the software (system software and application software) which is necessary to realize the data processing functions needed by its users.</p>
3.2 User	<p>This is a person (or instance) who uses the functions of a CBSS via a terminal (or an equivalent machine-user-interface) by initiating tasks and receiving the computed results.</p>
3.3 Workload (WL) and workload parameter set (WPS)	<p>The workload (WL) is the set of all users of a defined data processing system including the installed applications. I. e. the WL is the user entity of a CBSS.</p> <p>The WL is (see clause 5.1) defined by:</p> <ul style="list-style-type: none">- The workload parameter set (WPS) which defines the set of users and all of the essential properties of the users time behaviour.- All of the programs used by the users (including operating system command files as far as needed).- The list of all of the additionally needed software components (for instance program library, compiler, data base system,...).- Stored Data.
3.4 System under test (SUT)	<p>The system under test is the CBSS to be tested. I. e. it is the data processing system consisting of hardware, system software, data communication features and application-SW.</p>

Term (abbreviation in brackets)	Definition
3.5 Remote terminal emulator (RTE)	The remote terminal emulator is a data processing system emulating an user entity as defined by a WL (see above).
3.6 User type (UT)	It is assumed that the user entity can be classified into n_{Types} user types. Each user type is (see clause 5.1.1) described by three sets of characteristics: Relative frequencies of using task chains, mean think times, standard deviations of the think times.
3.7 Task	<p>An order submitted to the system under test demanding the execution of a data processing operation according to a defined rule (i. e. a part of an application program) how to produce output data from input data and (if needed) stored data.</p> <p>The task is defined by:</p> <ul style="list-style-type: none"> - The input information which has to be submitted to the system under test ("input string"). - The demanded run time (defined by a so called timeliness function TF, see subclause 3.15) - the so called jobmode JM (see subclause 3.13) which defines if the task will be processed in batch mode or in dialog mode.
3.8 Task type	<p>A set of types of tasks has to be defined. The emulated users submit only these types of tasks to the SUT. As described in subclause 5.1.1.2 for all of the task types</p> <ul style="list-style-type: none"> - the input string ("input string" see also "task", subclause 3.7) which has to be submitted to the system under test, - the timeliness function TF (TF see subclause 3.15) and - the value $M(j)$ of the job mode (job mode see subclause 3.13) have to be defined. <p>The current number of the task type is j. The total number of task types is m.</p>

Term (abbreviation in brackets)	Definition
3.9 Task chain	Due to the logic of the used application programs it is sometimes necessary that - beginning with a certain task type - a number of tasks have to be submitted to the SUT in a defined order: Task chain.
3.10 Chain type	A set of types of task chains has to be defined. The emulated users submit only chains of these types to the SUT. Each chain type is defined (see subclause 5.1.1.3) by a sequence of task types. The total number of chain types is u.
3.11 Execution time	<p>This is the time which elapses between the task start and the completion of the task of an user. The "task start" is defined as the instant when the user has completed the submission of the "input string" (see above at "task type", subclause 3.8). The instant when all of the information which has to be generated by the SUT (i. e. the computational result) is received by the user is defined as the "completion of the task".</p> <p>This definition of execution time holds for interactive tasks as well as for tasks which will be executed in batch mode.</p>
3.12 Think time	The think time (corresponding to the next task of an user) is the time which elapses between the completion of the current task ("completion of the task" see above at execution time, subclause 3.11) and the task start ("task start" see above at execution time, subclause 3.11) of the next task of this user. This is valid if the current task represents a dialog job. In case of the current task being a batch job the think time (corresponding to the next task of this user) is the time which elapses between the task start of the current (batch-) task and the task start of the next task of this user.

Term (abbreviation in brackets)	Definition
---------------------------------	------------

3.13 Job mode (JM)

The job mode describes whether a task which is submitted to the SUT is intended (by the user) to be a batch job or a dialog job.

The JM has the value $M=1$ (WAIT) in case of a dialog task and $M=0$ (NO WAIT) in the case of a batch task. For each of the task types the value of the job mode has to be defined. The remote terminal emulator has - before it starts the next task of an emulated user - to look first of all for the job mode of this task. This is necessary in order to handle the start of the think time in an adequate way (compare to the explanations given for the definition of "think time", subclause 3.12). $M(j)$ is the mathematical symbol of the job mode corresponding to the j -th task type.

3.14 Relative chain type frequency

The relative frequency of using the l -th chain type by an user of the i -th type is $q(i,l)$. Each of the simulated users generates the chains in a random sequence. But the tasks within each of the chains are generated in a strict order according to the chain type definition.

3.15 Timeliness function (TF)

A timeliness function corresponding to each of the task types has to be defined. This TF describes the requirements of the users with respect to the execution times.

A timeliness function defines one or more time limits for the execution times of the corresponding task type. Additionally the TF defines the relative frequencies corresponding to each of these time limits.

These relative frequencies are the maximum percentage up to which execution times (which are less or equal to the corresponding time limit) are accepted.

The number of time classes of the TF corresponding to the j -th task type is $z_T(j)$; the time limits are $g_T(j,k)$ where j is the current number of the task type and k is the current number of the time limit; the maximum relative frequencies are $r_T(j,k)$. The timeliness function corresponding to the j -th task type therefore consists of $z_T(j)$ couples of numbers " $g_T(j,k)$ and $r_T(j,k)$ ".

3.16 Mean execution time

This is the mean value of all of the execution times of tasks of the j -th task type which were started within the rating interval. The mathematical symbol is $T_{ME}(j)$.

3.17 Throughput

The rate (i. e. the average number per time unit with respect to the rating interval) of all of the tasks of the j -th task type submitted to the SUT is the throughput corresponding to this task type. The mathematical symbol is $B(j)$.

3.18 Timely throughput

The rate (i. e. the average number per time unit with respect to the rating interval) of all of those tasks of the j -th task type the execution times of which are accepted with respect to the TF corresponding to the j -th task type is the timely throughput of this task type. The mathematical symbol is $E(j)$.

3.19 Throughput reference value

The minimum rate of all of the tasks of the j -th task type which is required by the user entity is: $B_{REF}(j)$. It can be computed from the workload parameter set (WPS) of the user entity (see subclause 9.1.2).

3.20 Mean execution time reference value

The maximum average execution time of tasks of the j -th task type which is accepted by the user entity is $T_{REF}(j)$. It can be computed from the workload parameter set (WPS) of the user entity (see subclause 9.1.1).

3.21 Throughput rating value for the j-th task type

This is - corresponding to the j-th task type - the quotient of the (actual) total throughput $B(j)$ to the throughput reference value $B_{REF}(j)$:

$$R_{TH}(j) = \frac{B(j)}{B_{REF}(j)} .$$

3.22 Mean execution time rating value for the j-th task type

This is - corresponding to the j-th task type - the quotient of the mean execution time reference value $T_{REF}(j)$ to the measured mean execution time $T_{ME}(j)$:

$$R_{ME}(j) = \frac{T_{REF}(j)}{T_{ME}(j)} .$$

3.23 Timeliness rating value for the j-th task type

This is - corresponding to the j-th task type - the quotient of the timely throughput $E(j)$ to the total throughput $B(j)$:

$$R_{TI}(j) = \frac{E(j)}{B(j)} .$$

4 Abbreviations and symbols

4.1 Abbreviations

Abbreviation	Term
CBSS	Computer-based software system
DPS	Data Processing System
EU	Emulated user
JM	Job mode
RTE	Remote Terminal Emulator
SUT	System under Test
TF	Timeliness Function
UT	User type
WL	Workload
WPS	Workload parameter set

4.2 Symbols

Symbol	Meaning
ALPHA	Confidence coefficient
$B(j)$	Throughput corresponding to the j -th task type
$B_{REF}(j)$	Reference value of throughput corresponding to the j -th task type
$d(j)$	Half width of the confidence interval corresponding to the j -th task type
$DELTA_h$	RTE correctness factor of mean think time
$DELTA_q$	RTE correctness factor of chain frequency
$DELTA_s$	RTE correctness factor of the standard deviations of the think times
$E(j)$	Timely throughput corresponding to the j -th task type
$f(l,k)$	Task type of the k -th task in the l -th chain type
$g_T(j,k)$	k -th time limit of the timeliness function corresponding to the j -th task type
$h(i,j)$	Mean think time (before submitting a task) of the users of the i -th user type corresponding to the j -th task type

$L_{Chain(l)}$	Length of the l-th chain type
m	Number of task types
$M(j)$	Job mode corresponding to the j-th task type
$n(i)$	Number of users of the i-th user type
n_{tot}	Total number of users to be emulated by the RTE
n_{Types}	Number of user types
$q(i,l)$	Relative chain frequency: the relative frequency of using the l-th chain type by an user of the i-th user type.
$r_T(j,k)$	Maximum accepted relative frequency of tasks of the j-th task type whose execution times are less or equal to the time limit $g_T(j,k)$
$R_{ME(j)}$	Mean execution time rating value corresponding to the j-th task type
$R_{TH(j)}$	Throughput rating value corresponding to the j-th task type
$R_{Tl(j)}$	Timeliness rating value corresponding to the j-th task type
$s(i,j)$	Standard deviation of the think time of users of the i-th user type tasks of the corresponding to j-th task type
t_0	Start time of the RTE
t_1, t_2	Begin time and end time of the rating interval
t_3	End time of the supplementary run
$t_{ET(j,k)}$	Execution time of the k-th task (of j-th task type) which was started within the rating period
$T_{ME(j)}$	Mean execution time corresponding to the j-th task type
T_R	Duration of the rating interval. It holds: $T_R = t_2 - t_1$
$T_{REF(j)}$	Reference value of mean execution time corresponding to the j-th task type
u	Number of chain types
$x_{ET(j)}$	Total number of tasks of the j-th task type which were started within the rating period
$z_T(j)$	Number of time limits of the timeliness function corresponding to the j-th task type

5 Input needed for Measurement

5.1 Workload Description

5.1.1 The workload parameter set (WPS)

The user entity is described by a set of characteristics. The following values shall be defined:

5.1.1.1 Basic Parameters

- n_{Types} : Number of user types (user type shortened as UT)
- $n(1), n(2), \dots, n(n_{Types})$: Numbers of users of the types 1 to n_{Types}
- m : Number of task types
- u : Number of chain types

5.1.1.2 Definitions of the task types

For each of the m task types the following list of information or values respectively has to be defined (j is the current number of the task type):

- The input information which has to be submitted to the system under test ("input string"). If there are variations needed see subclause 5.3 .
- Timeliness function (TF)
It consists of the following data:
 - x) The number of time classes $z_T(j)$;
 - x) The upper time class boundaries $g_T(j, 1)$ to $g_T(j, z_T(j))$
 - x) The maximum relative frequencies $r_T(j, k)$ to $r_T(j, z_T(j))$
- The value of the job mode $M(j)$ (the value is 1, i. e. WAIT or 0, i. e. NO WAIT)
- The computation result of the task in case of correct operation of the SUT (compare subclause 5.3).

5.1.1.3 Definitions of the task chain types

For each of the u chain types the following values have to be defined (l is the current number of the chain type):

- $L_{Chain}(l)$ the length of the chain, i. e. the number of tasks which are contained in the sequence
- the current numbers of the task types in exactly the sequence in which these tasks shall be processed: $f(l,1)$ to $f(l, L_{Chain}(l))$, where $f(l,k)$ is the current task type number of the k -th task in this sequence.

5.1.1.4 Relative chain frequencies

For each of the n_{Types} user types the relative frequency of using the l -th chain type shall be defined. Therefore it is necessary to define the following n_{Types} rows each having u values (u represents the number of chain types):

User type 1:
 $q(1,1)$ to $q(1,u)$

User type 2:
 $q(2,1)$ to $q(2,u)$

.
.
.

n_{Types} -th user type:
 $q(n_{Types},1)$ to $q(n_{Types},u)$

where $q(i,l)$ is the relative frequency by which the l -th chain type is used by an user of the i -th user type.

Remarks:

- 1) If a task chain of a certain chain type (lets say the l -th type) do not occur in the job stream of users of a certain user type (lets say the i -th type) then $q(i,l)$ it has the value zero.
- 2) Due to the axiom that the sum of all relative frequencies with respect to all possible events has to be one it is valid for all of the user types:
The sum of $q(i,1)$ to $q(i,u)$ equals 1.
This may be used as a control if the stated q -values do not contain theoretical impossibilities.
- 3) If a task is intended to occur "single", i. e. it shall be not occur within a chain then, this can be realized as follows: Define a chain type of length 1 containing only this task type.

5.1.1.5 Mean think times and their standard deviations

For each of the n_{Types} user types the following list of think time mean values $h(i,j)$ and the corresponding standard deviations $s(i,j)$ has to be defined; i is the current number of the user type and j is the current number of the task type.

User type 1:
 $h(1,1)$ to $h(1,m)$;
 $s(1,1)$ to $s(1,m)$.

User type 2:
 $h(2,1)$ to $h(2,m)$;
 $s(2,1)$ to $s(2,m)$.

.
.
.

n_{Types} -th user type:
 $h(n_{Types}, 1)$ to $h(n_{Types}, m)$;
 $s(n_{Types}, 1)$ to $s(n_{Types}, m)$;

5.1.2 The (application) programs

All of the programs used by the users (including operating system command files as far as needed) shall be presented on a magnetic tape or an equivalent storage medium. These programs shall have the final form (full in source code or a compiled deck) ready to be used on the SUT.

5.1.3 The list of additionally needed software for the measurement run

All of the additional software components or standard system software modules (for instance system programs, libraries, compilers, data base system,..) which are needed to run the test have to be listed by exact name, version, release, distributor or manufacturer.

5.1.4 The stored data

All data which are needed by the programs as far as they are not contained in the input strings of the task type descriptions shall be presented. These data shall be presented completely on a magnetic tape or an equivalent storage medium and in the final form and ready to use so that they can be stored without any modification on the system under test. Examples for such data are:

- basic data files for computation
- the basic data of a data base system.

5.2 Description of the system under test (SUT)

5.2.1 Hardware architecture and configuration

These properties of the SUT shall be described comprehensively.

The following list is an non-normative example.

- Description of the architecture, for instance
 - x) monoprocessor system,
 - x) multiprocessor system,
 - x) front-end processor (if used),
 - x) Client/server system,
 - x) type of communication network,
 - x) types of enduser communication interfaces
(alphanumeric/graphic terminal, voice input/output,...)

- Hardware description (for instance in case of a conventional system architecture)
 - x) Manufacturer, distributor
 - x) CPU model, serial number, date of shipping
 - x) number of processors
 - x) size of main memory (Mbytes)
 - x) type, size of instruction/data cache
 - x) configuration of I/O-channels
 - x) secondary storage configuration
(number of disks, type, controllers, disk cache type and size, solid state disks - if used,)
 - x) mass storage (optical, magnetic,...)
 - x) front end configuration and terminal network configuration

5.2.2 System software configuration

The set of these software components shall be described comprehensively. At least the following information shall be listed:

Manufacturer, type and release

- of the operating system

- of the job scheduler

- of additional system components as far as used, i. e. software cache for disks.

- of compiler for each used programming language; additionally the used run-time system shall be listed, shared mode if used, used optimizer options, data compression features.

- of the data base system (if used); additionally the data configuration shall be listed.

5.2.3 Application software configuration

This set of software was mentioned in subclause 5.1.2. It shall be comprehensively specified in an analogous manner as it was stated and documented above for the system software.

5.3 The correct output of the computations in the measurement run

The result of the computation in case of correct operation of the system under test shall be listed for all of the task types. If there is any use of job modification (i. e. running numbers in case of opening accounts) then

- all rules of job modification shall be defined comprehensively and uniquely
- all modified results and or variations of job output shall be listed.

6.1 Steps of the measurement procedure

Step 1 Install the SUT (including terminal network and other communication features). Its hardware and system software shall be exactly the same as specified in the subclauses 5.2.1 and 5.2.2. The application software shall be exactly as specified in the subclauses 5.1.2 and 5.2.3.

Step 2 Install a RTE which is constructed according to the RTE specifications of this international standard and which is proven to fulfil these specifications.

Step 3 Connect SUT and RTE using the data communication features as installed in step 1.

Step 4 Load the WPS (see subclause 5.1.1), the set of all of the input strings (if necessary also the modification rules, compare subclause 5.3), and the set of correct outputs (for use of controlling the SUT's correct computation) into the RTE.

Step 5 Load the stored data (as specified in subclause 5.1.4) into the SUT.

Step 6 Now is the latest possible moment at which

- the values of the criteria for statistic validity of the results (see subclause 10.3) which are
 - x) the confidence coefficient ALPHA of mean response time and
 - x) the m confidence intervals $2d(j)$ of mean response times of all of the task types

as well as

- the values of the criteria for sufficient precise working of the RTE (see subclause 10.1) which are
 - x) DELTA_h
(this is the maximum accepted relative difference of the actual think time mean values to the defined values $h(i,j)$)
 - x) DELTA_s
(this is the maximum accepted relative difference of the think time's actual standard deviations to the defined values $s(i,j)$)
 - x) DELTA_q
(this is the maximum accepted relative difference of the actual relative chain frequencies to the defined values $q(i,l)$)

shall be stated.

Step 7 Define the length T_R of the so called rating interval. This is (see subclause 3.1.3.1 of part 1 of this international standard) the time from the end of the "warm up phase" to the beginning of the "supplementary run".

Step 8 Start running the R I E (this is time t_0) and wait for the end of the "warm up time", i. e. until a steady state is reached (there is no rule for the duration of this period).

Step 9 Start the rating interval. This is time t_1 .

Step 10 Start recording the logfile at t_1 . For each task
x) the log record (format see subclause 7.1) and
x) the task's complete output (for use of controlling the correct work of the SUT)
shall be stored.

Step 11 At the time $t_2 = t_1 + T_R$ the rating interval is finished. But the RTE's run shall still go on in full operation to t_3 .
 t_3 is the moment when all those tasks with task starts before or at t_2 ("Supplementary run") are completed.
 t_3 is the end of the supplementary run. Now the recording of the logfile may be stopped.

Step 12 At t_3 or later the RTE may be halted.

Step 13 Save the logfile and the computation result file.

Remark:

There is an observation period which begins at t_1 and ends at t_3 . All tasks the task start of which falls into the observation period are "within recording". But only those tasks the start of which falls into the rating period are taken into account for computing performance values. All the tasks the task start of which falls into the supplementary run are out of the rating. But the recorded information of the tasks which were started within the supplementary run is needed for controlling the correctness of the measurement.

6.2 Modification of the rating interval definition

For reasons of easier practicability the following modification is allowed: If at time t_2 (the end of the rating interval) and/or t_3 (the end of the supplementary interval) the last task of an emulated user is not the last task of the running chain of this user then a later "local interval end" t_{2loc} or t_{3loc} respectively may be used for this user in order to complete the running chain. Such local interval ends also may be used if this would improve the accuracy (compare subclause 10.1 of part 2 of this international standard) of the work of the RTE i. e. improves the values of the indicators $DIFF_h$, $DIFF_s$ and $DIFF_q$. But the following rules shall be fulfilled:

- The length of a local rating interval is at maximum 110% of the planned length T_R of rating interval.
- The lengths of the local supplementary runs differ at a maximum of 110%.
- The RTE's run goes on in full operation for all of the emulated users to the end of the last local supplementary run. This time is named t_4 . It is the moment when all those tasks are completed whose task starts had occurred within any one of the local supplementary runs. The recording of the logfile shall not be stopped at t_3 but shall be continued until t_4 .

7.1 Measurement logfile

This is the file containing all of the logfile records according to step 10 in clause 6. Each logfile record contains the following information for the corresponding task:

- Current number of the task (chronological since t_1)
- Type of the user which submitted the task and the individual identification of this user
- Task type
- Type of the task chain in which the task is contained
- Serial number of the task within the chain
- Time when the think time started for this task (typical precision: 1/100 sec)
- Time of task start (typical precision: 1/100 sec)
- Time of completion of this task (typical precision: 1/100)

Remark:

The cited values of precision for registered times are recommended values for online transactions and timesharing commands. If there are other classes of tasks (for instance information transport in a network or real-time functions) then proper values (which are typically shorter than 1/100 sec) can be stated.

7.2 Computation result file

For use of controlling the correct work of the SUT all computation results of the observation period (rating interval plus supplementary run) shall be stored in the so called computation result file. That means:

- The computed output which submits the SUT to the users has to be recorded (and stored in the computation result file) for all of the tasks the task start of which is after t_1 .
- The "stored data" (see subclause 5.1.4) have to be recorded at t_3 (and stored in the computation result file).

Remark:

In case of the SUT being a large CBSS the computed output which is submitted by the SUT to the user entity can be a large amount of data. If a problem arises in storing these data the following alternative solution may be chosen: The RTE verifies each the of task responses immediately to ensure it is correct. Only those outputs which are not correct have to be stored.

8 Estimation of Performance values

The duration of the rating period (compare the remark at the end of clause 6.1) is

$$T_R = t_2 - t_1 \quad .$$

8.1 Throughput

This is the average number of all of the tasks of the j-th task type submitted to the SUT within the rating period per time unit:

$$B(j) = \frac{\text{Number of tasks of the j-th type, started within the observation period}}{T_R}$$

B(j) shall be computed for all of the task types, i. e. for $j=1,2,\dots,m$
(Remark: A program to perform this procedure can be found in annex F).

8.2 Mean execution time

This is the average value $T_{ME}(j)$ of the execution time of all tasks of the j-th task type which were submitted to the SUT within the rating period:

$$T_{ME}(j) = (t_{ET}(j,1) + \dots + t_{ET}(j,k) + \dots + t_{ET}(j,x_{ET}(j))) / x_{ET}(j)$$

where $x_{ET}(j)$ is the total number of tasks of the j-th task type submitted to the SUT within the rating period

and $t_{ET}(j,k)$ is the duration of the execution time of the k-th task (of the j-th task type) submitted to the SUT within the rating period

$T_{ME}(j)$ shall be computed for all of the task types, i. e. for $j = 1, 2, \dots, m$.
(Remark: A program to perform this procedure can be found in annex F).

8.3 Timely throughput

This is the average number $E(j)$ of all those tasks of the j -th task type per time unit which are accepted as having been completed in time. "Accepted as being in time" means "being completed in time with respect to the timely function which corresponds to the j -th task type".

$$E(j) = \frac{\text{Number of timely completed tasks of the } j\text{-th task type,} \\ \text{the task start of which is within the rating period}}{T_R}$$

$E(j)$ shall be computed for all of the task types, i. e. for $j = 1, 2, \dots, m$ (Remark: The procedure showing how to compute $E(j)$ is somewhat sophisticated. It is described in annex B.3. A program to compute $E(j)$ can be found in annex F.)

9 Rating of the measured performance values of the SUT

9.1 Computation of the performance reference values

9.1.1 Mean execution time reference values

For each of the task types there is a reference value:

$$T_{REF}(1), T_{REF}(2), \dots, T_{REF}(m)$$

$T_{REF}(j)$ is (according to the timeliness function of the j -th task type) the maximum accepted mean execution time for tasks of the j -th task type.

$T_{REF}(j)$ can be computed from values which are defined in the workload parameter set (WPS). The m mean execution time reference values shall be computed.

The computation is explained in clause 3.4.1.1 of part 1 of this international standard. The formula is described in annex B.1.

Remark:

A program to perform this procedure can be found in annex F.

9.1.2 Throughput reference values

For each of the task types there is a reference value:

$$B_{REF}(1), B_{REF}(2), \dots, B_{REF}(m)$$

$B_{REF}(j)$ is the number of tasks of the j -th task type per time unit which is - as a the minimum value - required by the user entity.

$B_{REF}(j)$ can be computed from values which are defined in the workload parameter set (WPS). The m throughput reference values shall be computed.

Remark:

The formula showing how to compute $B_{REF}(j)$ is somewhat sophisticated. It is described in annex B.2. The source of a short running program to compute $B_{REF}(j)$ can be found in annex F.

9.2 Computation of the rating values

9.2.1 Computation of the throughput rating values

The throughput rating values for all of the task types i. e.

$$R_{TH}(1), \dots, R_{TH}(j), \dots, R_{TH}(m)$$

shall be computed. $R_{TH}(j)$ is (for the j-th task type) the quotient of the actual throughput $B(j)$ to the throughput reference value $B_{REF}(j)$:

$$R_{TH}(j) = \frac{B(j)}{B_{REF}(j)}$$

Remark:

A program to perform this computation can be found in annex F.

9.2.2 Computation of the execution time rating values

The mean execution time rating values for all of the task types i. e.

$$R_{ME}(1), \dots, R_{ME}(j), \dots, R_{ME}(m)$$

shall be computed. $R_{ME}(j)$ is (for the j-th task type) the quotient of the mean execution time reference value $T_{REF}(j)$ to the measured mean execution time $T_{ME}(j)$:

$$R_{ME} = \frac{T_{REF}(j)}{T_{ME}(j)}$$

Remark:

A program to perform this computation can be found in annex F.

9.2.3 Computation of the timeliness rating values

The timeliness rating values for all the task types i. e.

$$R_{TI}(1), \dots, R_{TI}(j), \dots, R_{TI}(m)$$

shall be computed. $R_{TI}(j)$ is (for the j-th task type) the quotient of the timely throughput $E(j)$ to the measured (total) throughput $B(j)$:

$$R_{TI}(j) = \frac{E(j)}{B(j)}$$

Remark:

A program to perform this computation can be found in annex F.

9.3 Rating

9.3.1 Rating in detail

9.3.1.1 Throughput rating

If the throughput rating value of the j -th task type $R_{TH}(j)$ is less than 1 then the throughput of the CBSS corresponding to this task type is poor.

If the throughput rating value of the j -th task type $R_{TH}(j)$ equals 1 or is greater than 1 then the throughput of the CBSS corresponding to this task type is good.

This rating shall be done for all m task types.

9.3.1.2 Mean execution time rating

If the mean execution time rating value of the j -th task type $R_{ME}(j)$ is less than 1 then the mean response time of the CBSS corresponding to this task type is poor.

If the mean execution time rating value of the j -th task type $R_{ME}(j)$ equals 1 or is greater than 1 then the mean execution time of the CBSS corresponding to this task type is good.

This rating shall be done for all m task types.

9.3.1.3 Timeliness rating

If the timeliness rating value of the j -th task type $R_{TI}(j)$ is less than 1 then the timeliness of the CBSS corresponding to this task type is poor.

If the timeliness rating value of the j -th task type $R_{TI}(j)$ equals 1 then the timeliness of the CBSS corresponding to this task type is good.

This rating shall be done for all m task types.

Remark:

The timeliness rating values $R_{TI}(j)$ can not exceed the value 1. The reason is that as a maximum all of the tasks are completed in time, but not more. Therefore it is not necessary to look for values of $R_{TI}(j)$ greater than 1 or to try to rate such values.

9.3.2 Overall rating

If at least one of the $3 \cdot m$ rating values (i. e. m values $R_{TH}(j)$ plus m values $R_{ME}(j)$ plus m values $R_{TI}(j)$) is less than 1 then the CBSS does not fulfil at least one criteria of the requirements of the user entity. Therefore the CBSS has to be rated as being poor. Otherwise it works sufficiently according to all the time behaviour requirements of the user entity and the CBSS shall be rated as being good.

Remark:

This standard gives no recommendation on how to handle the situation if one or more rating values is/are very large (compared to the value 1) because no general rule exists on how to rate a situation where the user requirements are more than fulfilled.

10 Validation of Measurements

General rule: After having finished the measurement experiment (i.e. the steps 1 to 13 according to subclause 6.1 are executed) the validations according to the subclauses 10.1 to 10.3 of this part 2 of the standard shall be done.

10.1 Validation of the accuracy of the RTE

Step 1

The logfile (see step 13 in clause 6.1) has to be analysed. The actual relative chain frequencies of all those chains the start of which was between time t_1 and t_3 (rating interval plus supplementary interval) shall be computed. This shall be done separately for each user. Then for all of these "computed actual relative chain frequencies" the differences to the values $q(i,l)$, as they are defined in the WPS, shall be computed. The computations may be done by a program. Formula see annex B.4.1.

The computed relative chain frequency differences have the mathematical symbol $DIFF_q$. If there holds

$$DIFF_q \leq DELTA_q$$

separately for all of the emulated users and for all of the task types

then the accuracy of the RTE with respect to the chain generation was sufficient.

Otherwise the RTE did not work precisely enough and the measurement results are not acceptable. A new measurement using an improved RTE shall be done.

Step 2

Analogously to step 1 the entries in the recorded logfile (see step 13 in clause 6.1) concerning the time t_1 to t_3 have to be analysed with regard to the think times. The "actual" mean values and the "actual" standard deviations of the think times shall be computed. This shall be done separately for each one of the user groups (for all of those tasks which were submitted between t_1 and t_3) and separately for all of the task types used by the group. (An user group is the set of all emulated users of the same user type.) Then for all of these "computed actual mean values" and "computed actual standard deviations" the relative differences to the values $h(i,j)$ and $s(i,j)$, as they are defined in the WPS, shall be computed. The computations may be done by a program. Formula see annexes B.4.2 and B.4.3.

The computed relative think time differences have the symbol $DIFF_h$ and the computed relative standard deviation differences have the symbol $DIFF_s$. If there holds

$$DIFF_h \leq DELTA_h \quad \text{and} \quad DIFF_s \leq DELTA_s$$

separately for all of the emulated users and for all of the task types

then the accuracy of the RTE with respect to the think time generation was sufficient.

Otherwise the RTE did not work precisely enough and the measurement results are not acceptable. A new measurement using an improved RTE has to be done.

10.2 Validation of the computation correctness of the SUT

The recorded computation result file (see step 13 in clause 6.1) has to be analysed. For all of the recorded tasks the results computed by the SUT have to be compared to the defined "correct values" (see last dash in subclause 5.1.1.2 and subclause 5.3). This may be done by a proper program. If all of the task's results were complete and proven to be correct then the measurement is acceptable with respect to the correctness of the SUT.

Otherwise the SUT did not work correctly. A new measurement shall be done.

10.3 Validation of the statistical significance of the measurement results

For each one of the task types a test concerning the statistical significance of the estimated mean execution time values shall be performed.

The sequential test as defined in annex B.6 shall be used. The test examines - for the j -th task type - the $x_{ET}(j)$ values $t_{ET}(j,1)$, $t_{ET}(j,2), \dots, t_{ET}(j, x_{ET}(j))$ which were used (see subclause 8.2) for the computation of the mean execution time. The test refer to the defined value of the confidence coefficient ALPHA. Additionally the test refer to the confidence intervals $d(j)$ which are defined separately for each of the task types. The test result is delivered in a variable named SEQTEST. The result values are: "OK" or "NOT OK" respectively.

If there holds

SEQTEST = OK for all of the m task types

then the statistical significance of the measurement result is sufficient.

Otherwise one or more of the estimated mean values and, in consequence, one or more of the estimated performance values are not sufficiently significant. They are not acceptable. A new measurement shall be done.

Specification of the RTE's basic functions

The RTE emulates a number of n_{tot} users. This number is the sum of the numbers of users of type 1, of type 2,.....:

$$n_{\text{tot}} = n(1) + n(2) + \dots + n(n_{\text{Types}}) .$$

All of the users shall be emulated independently. An emulated user (EU) works as follows:

The EU submits chains of tasks to the SUT. The sequence of the tasks corresponds to the chosen task chain type (see subclause 5.1.1.3). To start his work the EU randomly chooses a chain type. Let us assume that the regarded EU is one of the i -th type and that the number of the chosen chain is l_1 . Additionally let us assume that the relative chain frequency $q(i,l_1)$ does not equal zero. Then the EU submits tasks according to the task type sequence in the definition of the l_1 -th chain type. Having submitted this sequence the EU randomly chooses a new chain type. Let us assume that the number of the chain type is l_2 and that the relative chain frequency $q(i,l_2)$ does not equal zero. Then the SE submits tasks according to the task type sequence in the definition of the l_2 -th chain type. And so on. The chain types shall be chosen randomly by the i -th EU but according to the defined values of the relative chain frequencies $q(i,1)$, $q(i,2)$, , $q(i,u)$.

The submission of the tasks of a chain shall be as follows: The EU waits for a (randomly taken) think time and submits a task corresponding to the first task type in the chain definition (see subclause 5.1.1.3). Let us assume that this first task type is j_1 and the next task type is j_2 . Then the EU considers the job mode $M(j_2)$ of the j_2 -th task type. If $M(j_2)$ equals zero (i. e. batch mode) the EU waits for a (randomly taken) think time and then submits a task of the j_2 -th task type. If the above mentioned $M(j_2)$ equals 1 (dialog mode) then the EU waits for the output of the first task; when having received this output completely the EU waits for a (randomly taken) think time after which it submits the 2nd task. And so on. Having submitted the last task of the chain, the EU chooses (randomly but corresponding to the relative chain frequencies) a new chain type. The task type of the first task of this chain and the job mode of this task are considered. If the value of the job mode is 1 the EU waits for the output of the running task (i. e. the last task of the just finished chain) before starting the think time of the next chain's first task.

If the value of the job mode is zero the EU does not wait for the output of the running task (i. e. the last task of the just finished chain); the think time of the next chain's first task begins at once.

The duration of think times shall be taken randomly but according to the thinktime mean values $h(i,1)$, $h(i,2)$, , $h(i,m)$ and standard deviation values $s(i,1)$, $s(i,2)$, , $s(i,m)$ as they are defined in subclause 5.1.1.5.

i is the type of EU and j is the type of the submitted task which follows the considered think time.

Additional formula/methods of computation

B.1 Formula of mean execution time reference value $T_{REF}(j)$

$$T_{REF}(j) = \sum_{k=1}^{Z_{\tau}(j)} g_{\tau}(j,k) \cdot [r_{\tau}(j,k) - r_{\tau}(j,k-1)]$$

For the meaning of the mathematical terms see subclause 4.2.

B.2 Formula of throughput reference value $B_{REF}(j)$

$$B_{REF}(j) = \sum_{i=1}^{n_{Types}} \left[\frac{n(i) \cdot \sum_{l=1}^u \{q(i,l) \cdot a(j,l)\}}{\sum_{l=1}^u \left\{ q(i,l) \cdot \left[\sum_{v=1}^{L_{Chain}(l)} h(i,f(l,v)) \right] + \left[\sum_{v=1}^{L_{Chain}(l)-1} (T_{REF}(f(l,v)) \cdot M(f(l,v+1))) \right] + T_{REF}(f(l,L_{Chain}(l))) \cdot M^*(i) \right\}} \right]$$

For the meaning of the mathematical terms used in this formula see subclause 4.2. Additionally

$$A) \quad M^*(i) = \sum_{l=1}^u [q(i,l) \cdot M(f(l,1))]$$

B) $a(v,l)$ is the number of tasks of the v -th task type within a chain of the l -th chain type.

is defined.

B.3 Algorithm for computing the timely throughput $E(j)$

Let $b(j) = B(j) \cdot T_R$. This value is the number of tasks of the j -th type which are completed by the SUT within the rating interval. All of the tasks of the j -th type are included, regardless of the user that generated them. The TF stipulates how many of these tasks are to be considered as completed in time.

The number of completed tasks in the k -th time class is named $e_x(j,k)$.

By definition is:

$$e_x(j,0) = 0.$$

$e_x(j,k)$ for $k > 0$ is defined by the following recursive formula:

$$e_x(j,k) = \min \left[\text{No} \left(\text{TR} \left| t_{ET}(j,k') \leq g_T(j,k) \right. \right) - \sum_{l=0}^{k-1} e_x(j,l) \text{ AND } (r_T(j,k) - r_T(j,k-1)) \cdot b(j) \right]$$

for $k > 0$ and k' running in the interval $0 < k' < \infty$.

In this formula $\min[a_1 \text{ AND } a_2]$ is a function which chooses the smaller value from the couple a_1, a_2 . The function $\text{No}(\text{TR}|\text{COND})$ counts the number of tasks of the j -th type, that fulfil the condition COND.

Explanation of this formula:

The maximum number of completed tasks of the k -th time class, is defined by the product of the relative class frequency, $r_T(j,k) - r_T(j,k-1)$, and $b(j)$. This is the right-hand term in the min function. The left-hand term in the min function takes the number of task requests whose execution time does not exceed the k -th time class boundary. However, not all these execution times contribute to the number of completed tasks in this time class. On the contrary, the number of execution times which have already been accounted for in the time classes 1, 2, 3, ..., $k-1$ have to be subtracted. The left-hand term may be smaller than, equal or greater than the right-hand term. The min function takes, if unequal, the smaller term. This is the number of completed tasks of the k -th time class.

The number of completed tasks of the j -th task type is

$$e(j) = \sum_{k=1}^{z_T(j)} e_x(j,k)$$

The number of completed tasks of each task type is the vector of timely completed tasks:

$$e = \begin{bmatrix} e(1) \\ e(2) \\ \bullet \\ \bullet \\ \bullet \\ e(m) \end{bmatrix}$$

Dividing this vector by the rating period T_R yields the vector of achieved performance:

$$E = \begin{bmatrix} E(1) \\ E(2) \\ \bullet \\ \bullet \\ \bullet \\ E(m) \end{bmatrix}, \text{ where } E(j) = \frac{e(j)}{T_R}$$

B.4 Formula for computing the indicators of the RTE's precision of work

B.4.1 Formula DIFF_q

$$\text{DIFF}_q(i,l) = \frac{q_{\text{measured}}(i,l) - q(i,l)}{q(i,l)}$$

B.4.2 Formula DIFF_h

$$\text{DIFF}_h(i,j) = \frac{h_{\text{measured}}(i,j) - h(i,j)}{h(i,j)}$$

B.4.3 Formula DIFF_s

$$\text{DIFF}_s(j) = \frac{s_{\text{measured}}(i,j) - s(i,j)}{s(i,j)}$$

B.5 Formulas for mean value and standard deviation

Mean value mv

Standard deviation sd

$$mv = (1/Z) \cdot \sum_{k=1}^Z z(j)$$

$$sd = \sqrt{(1/Z) \cdot \sum_{k=1}^Z [z(j) - mv]^2}$$

where Z is the number of samples and z(1), z(2), ... , z(Z) are the samples.

B.6 The sequential test

To decide whether the confidential intervals are sufficient or not, a sequential test is used. It estimates the minimum number $N(j)$ of execution time samples $t_{ET}(j,k)$ for the j -th task type. Indirectly, the minimum length of the observation period is also defined hereby.

The sequential test may be done simultaneously with the measurement procedure or after it. The following algorithm determines whether or not (i.e. is a test stop criterion) the number of samples is sufficient for the j -th task type.

- (1) Take a first sample $t_{ET}(j,k)$.
- (2) Take another sample $t_{ET}(j,k)$.
- (3) Compute the mean value

$$t_{AN}(j) = 1/N(j) \cdot \sum_{k=1}^{N(j)} t_{ET}(j,k)$$

and the square of the standard deviation

$$\text{var}(j) = 1/N(j) \cdot \sum_{k=1}^{N(j)} [t_{ET}(j,k) - t_{AN}(j)]^2$$

- (4) If the inequality $N(j) < (u(\alpha/d(j)))^2 \cdot \text{var}(j)$ holds then go back to step 2. In this inequality „ $u(\alpha)$ “ is the $u_{1-\alpha/2}$ - quantile of the standard normal distribution „ $N(0,1)$ “ and is defined by:

$$\frac{2}{\sqrt{2\pi}} \cdot \int_0^{u(\alpha)} e^{-x^2/2} dx = 1 - \alpha$$

α is the confidence coefficient ALPHA.

It is not necessary to carry out the computation of the values $u(\alpha)$ by use of this difficult equation.

The values of $u(\alpha)$ are tabulated in most statistic handbooks and may be taken from there.

- (5) If the stop criteria holds for all of the execution times, then the length of the rating period is sufficient.

Standardized format of the workload description

The workload consists of 7 parts (C.1 to C.7):

C.1 Workload parameter set (WPS)

C.1.1 Basic parameter values

This list of the basic parameters has 6 entries:

- number of user types: n_{Types}
- numbers of users of each type: $n(1), n(2), \dots, n(n_{Types})$
- number of operation types contained in the task types: w
- number of task types: m
- number of chain types: u
- number of different timeliness functions: n_{TF}

C.1.2 Task type definitions

This list has m entries each consisting of four values:

- current number of the task type
- number of the operation type used in this task type
(for "operation type" see subclause C.2)
- Value of the job mode M
- type number of the timeliness function

C.1.3 Definitions of the timeliness functions

This list has n_{TF} entries each consisting of the following values:

- type number of the timeliness function
- number of time limits of this timeliness function: z_T
- z_T couples of the values g_T and r_T , where g_T is the time limit and r_T is the maximum accepted relative frequency (compare subclause 3.15 of part 2 of this international standard).

C.1.4 Definitions of the chain types

This list has u entries each consisting of the following values:

- current number of the chain type
- length of the chain : L_{Chain}
(This is the number of tasks within this chain)
- the sequence of the L_{Chain} type numbers of the tasks contained in this chain

C.1.5 Definitions of the chain probabilities

This is a matrix of u rows and n_{Types} columns. The element in the l -th row and i -th column is the relative frequency with which an emulated user of the i -th type initiates task chains of the l -th type. The sum of the elements of each column shall equal 1.

C.1.6 Think time mean values

This is a matrix of m rows and n_{Types} columns. The element in the j -th row and i -th column is the mean think time of an emulated user of the i -th type before starting a task of the j -th type.

C.1.7 Think time standard deviations

This is a matrix of m rows and n_{Types} columns. The element in the j -th row and i -th column is the standard deviation of the think time of an emulated user of the i -th type before starting a task of the j -th type.

C.2 Operation type definitions

For each of the w operation types shall be defined:

- The logical meaning of the input
There are three types:
 - x) command to the operating system (usual computer operating system or network operating system) of the SUT,
 - x) name of a command procedure to be executed by the operating system of the SUT,
 - x) input to a running program.
- The length (number of characters) of the input string submitted by the emulated user or the number of graphic actions (for instance mouse moving steps etc.) .
- The input string itself or the list of the graphic actions (for instance mouse moving steps plus "click").
- If there is any use of job modification (i. e. running numbers in case of opening accounts) then all rules on how to modify the input strings or (in case of graphic input actions) the sequence of the graphic input steps shall be defined comprehensively and uniquely.

C.3 Application programs

All of the programs (compare subclause 5.1.2 of part 2 of this standard) used by the users shall be presented on a magnetic tape or an equivalent storage medium. These programs shall have the final form (full in source code or a compiled deck) ready to use on the SUT.

C.4 Operating system command procedures

For all of the operation types the input (submitted to the SUT) of which represents the name of a command procedure (to be executed by the operating system of the SUT) the command procedures shall be presented in the final form (full operating command listings). The command procedures shall be presented on a magnetic tape or an equivalent storage medium, ready for use.

C.5 Computational results

C.5.1 Results for fixed input

In many cases the input string (or the sequence of graphic actions respectively) for a task type is the same whenever a task of this type is initiated.

The result of the computation in case of correct operation of the SUT according to the input has to be listed for all of these task types.

C.5.2 Results for varied input

If there is any use of task or job modification (compare subclause C.2; for instance running numbers in case of opening accounts) then

- all modified results and or variations of output shall be listed

or alternatively

- all the rules of modification of the output shall be defined comprehensively and uniquely.

C.6 Basic data

All data which are needed by the programs as far as they are not contained in the input strings of the task type descriptions have to be presented. These data shall be presented completely on a magnetic tape or an equivalent storage medium and in the final form and ready to use so that they can be stored without any modification on the system under test. Examples for such data are:

- basic data files for computation,
- the basic data of a data base system.

C.7 Statistical parameters and accuracy parameters

This international standard demands validations of the results of a measurement and rating procedure (compare clause 10 of part 2). The standard defines the method on how to perform the validation. The standard does not define values for the validation criteria because it is not possible to define proper values in general. But with respect to a considered workload proper values can be defined. Therefore recommended values of these criteria in each workload should be listed.

C.7.1 Criteria for the statistic significance of the measurement

These criteria are (see subclause 10.3 of part 2 of this international standard) the confidence coefficient ALPHA of mean execution time and the m confidence intervals $2d(j)$ of mean execution times of all of the task types. The recommended value of ALPHA and the recommended m values of $d(j)$ shall be defined.

C.7.2 Criteria for sufficient precise working of the RTE

These criteria are (see subclause 10.1 of part 2 of this international standard):

- x) $DELTA_h$ (this is the maximum accepted relative difference of the actual think time mean values to the defined values $h(i,j)$)
- x) $DELTA_s$ (this is the maximum accepted relative difference of the think time's actual standard deviations to the defined values $s(i,j)$)
- x) $DELTA_q$ (this is the maximum accepted relative difference of the actual relative chain frequencies to the defined values $q(i,l)$)

The recommended values of $DELTA_h$, $DELTA_s$ and $DELTA_q$ shall be defined.

Standardized format of the logfile

For each of the tasks initiated within the observation period (i.e. from the time t_1 which is the beginning of the rating interval to the time t_3 which is the end of the supplementary run) a data set shall be recorded and stored in the logfile. The data set shall have (at least) the following nine entries:

- Current number of the task (chronologically counted for all of the tasks generated by the SUT since t_1)
- type of the emulated user who initiated the task
- the identification (a string or a number) of the emulated user which initiated the task
- type number of the task
- type of the chain within which the task is contained
- current number of the task within the chain in which the task occurred
- begin time of the think time preceding the task
- time of task start (this is the time when the input string is completely submitted to the SUT and the emulated user's signal for the task start was submitted)
- time of task completion (this is the time when the output string is completely submitted to the destination; the destination shall be defined; it may be the user and/or another instance)

Programs

E.1 Program TREFRE

This program computes the mean execution time reference values $T_{REF}(j)$ (formula see annex B.1).

The program is available in the programming languages:

C, FORTRAN, PASCAL.

Source code length: 2 pages

Run time: Some seconds (workstation)

The programs and an example for demonstration of its application are delivered on disk by ISO.

< Remark of the editor: As long as this project has the status working draft the program disk is available from the editor. In case of interest please contact the editor. >

E.2 Program BETARE

This program computes the throughput reference values $B_{REF}(j)$ (formula see annex B.2).

The program is available in the programming languages:

C. FORTRAN, PASCAL.

Source code length: 4 pages

Run time: Some seconds (workstation)

The programs and an example for demonstration of its application are delivered on disk by ISO.

< Remark of the editor: As long as this project has the status working draft the program disk is available from the editor. In case of interest please contact the editor. >

E.3 Program BANDE

This program computes the timely throughput $E(j)$ (formula see annex B.3).

The program is available in the programming languages:

C, FORTRAN, PASCAL.

Source code length: 4 pages

Run time: One second to some minutes (workstation)

The programs and an example for demonstration of its application are delivered on disk by ISO.

< Remark of the editor: As long as this project has the status working draft the program disk is available from the editor. In case of interest please contact the editor. >

E.4 Program DIFF

This program computes the indicator values $DIFF_q$, $DIFF_h$, $DIFF_s$ for the validation of the RTE's accuracy (formula see annex B.4 and subclause 10.1).

The program is available in the programming languages:

C, FORTRAN, PASCAL.

Source code length: 1 page

Run time: Several seconds (workstation)

The programs and an example for demonstration of its application are delivered on disk by ISO.

< Remark of the editor: As long as this project has the status working draft the program disk is available from the editor. In case of interest please contact the editor. >

E.5 Program MANDS

This program computes mean values and standard deviations (formula see annex B.5) which are needed in some clauses.

The program is available in the programming languages:

C, FORTRAN, PASCAL.

Source code length: 1 page

Run time: Several seconds (workstation)

The programs and an example for demonstration of its application are delivered on disk by ISO.

< Remark of the editor: As long as this project has the status working draft the program disk is available from the editor. In case of interest please contact the editor. >

E.6 Program SETEST

This program computes the values of the binary variables SEQTEST (algorithm see annex B.6 and subclause 10.3) which are needed for the validation of the statistical significance of the measurement results.

The program is available in the programming languages:

C, FORTRAN, PASCAL.

Source code length: 6 pages

Run time: Several seconds to several minutes (workstation)

The programs and an example for demonstration of its application are delivered on disk by ISO.

< Remark of the editor: As long as this project has the status working draft the program disk is available from the editor. In case of interest please contact the editor. >

The complete data of the workload examples are delivered on disk or tape by ISO.

Example 1: **SIMPLOAD1**

This is a simple workload which uses very simple operation types. It may be used

- x) to gain a better understanding of the method used in this standard,
- x) for functional test of an RTE

and so on.

This workload is written for SUTs using an operating system of UNIX system V.4 type. But it can be easily migrated to any other multi-user operating system. The application is written by use of the programming language C but it can be easily migrated to another programming language.

Example 2: **SIMPLOAD2**

This is a simple workload which uses very simple operation types. It may be used

- x) to gain a better understanding of the method used in this standard,
- x) for functional test of an RTE

and so on.

This workload has only one user type. The advantage of this is that the number n_{tot} of users can be modified by steps of 1. Among others this workload may be used to show the effect of varying the number of emulated users by steps of 1. By performing a series of measurements using increasing numbers of emulated users it is possible to determine the maximum number of timely served users characterized by this workload. This is the maximum number of users for which none of the 3-m rating values is below 1.

This workload is written for SUTs using an operating system of UNIX system V.4 type. But it can be easily migrated to any other multi-user operating system. The application is written by use of the programming language C but it can be easily migrated to another programming language.

Example 3: SIMPLOAD3

This workload differs from SIMPLOAD2 only in having application programs with less efficiency (but the same functionality). The effect of less runtime efficiency may be shown if comparing the measurement and rating results to those SIMPLOAD2.

This workload is written for SUTs using an operating system of UNIX system V.4 type. But it can be easily migrated to any other multi-user operating system. The application is written by use of the programming language C but it can be easily migrated to another programming language.

Example 4: COMPCENTER1

This is a workload which uses COBOL and FORTRAN application programs for operation types. It represents a job stream profile which is characteristic for computer centre operation: Programming users plus a heavy batch job stream.

This workload has only one user type. The advantage of this is that the number n_{tot} of users can be modified by steps of 1. Among others this workload may be used to show the effect of varying the number of emulated users by steps of 1. By performing a series of measurements using increasing numbers of emulated users it is possible to determine the maximum number of timely served users characterized by this workload. This is the maximum number of users for which none of the 3-m rating values is below 1.

This workload is written for SUTs using an operating system of UNIX system V.4 type. But it can be easily migrated to any other multi-user operating system.

Example 5: COMPCENTER2

This is a workload the application program of which is a simple OLTP system (OLTP means online transaction processing). It represents an user entity which is typical for OLTP computer centre operation. For reasons of simplicity there is no use of a database system. The application uses "flat files".

This workload has 2 user types. The ratio of the numbers of user types $n(1)$ to $n(2)$ is 1 to 4. Among others this workload may be used to show the effect of varying the number of emulated users. Multiplying both $n(1)$ and $n(2)$ - which are the numbers of the emulated users of type 1 or 2 respectively - by factors of 2, 3, 4... the total number of users increases by increments of 5. By performing a series of measurements using increasing numbers of emulated users the maximum number of timely served users can be estimated. This is the maximum number of users for which none of the 3-m rating values is below 1. In most cases there is at most no disadvantage to the fact that the number of users cannot be varied by steps of 1 but only by steps of 5. This is due to the fact that a CBSS typically serves a great number of users. Therefore the relative failure (of the estimated maximum number of timely served users) arising from the increase of users by steps of 5 is not too dramatic.

This workload is written for SUTs using an operating system of UNIX system V.4 type. But it can be easily migrated to any other multi-user operating system. The application is written by use of the programming language COBOL.

Example 6: COMPCENTER3

This workload differs from COMPCENTER2 only in having application programs with less efficiency (but the same functionality). The effect of less runtime efficiency may be shown if composing the measurement and rating results to those COMPCENTER2.

This workload is written for SUTs using an operating system of UNIX system V.4 type. But it can be easily migrated to any other multi-user operating system. The application is written by use of the programming language C but it can be easily migrated to another programming language.