
**Information technology — Machine
readable test data for biometric
testing and reporting —**

**Part 1:
Test reports**

*Technologies de l'information — Données d'essai lisibles par machine
pour les rapports et les essais biométriques —*

Partie 1: Rapports d'essai

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 29120-1:2015



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2015

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

	Page
Foreword	iv
Introduction	v
1 Scope	1
2 Conformance	1
3 Normative references	1
4 Terms and definitions	2
5 Symbols and abbreviated terms	2
6 ASN.1 format	2
6.1 Encoding rules	2
6.2 ASN.1 object identifier for test report	2
6.3 BiometricTestReport type	2
6.4 Data types for technology tests	3
6.4.1 Overview	3
6.4.2 Product information	3
6.4.3 Information about test report	5
6.4.4 Test report under a specific condition	7
6.5 Data types for scenario tests	12
6.5.1 Overview	12
6.5.2 Test report under a specific condition	12
6.6 Data types for signed test reports	13
Annex A (normative) ASN.1 module for machine readable biometric test reports	15
Annex B (informative) Common elements	22
Annex C (informative) Test reports	29
Bibliography	37

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT), see the following URL: [Foreword — Supplementary information](#).

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, SC 37, *Biometrics*.

ISO/IEC 29120 consists of the following parts, under the general title *Information technology — Machine readable test data for biometric testing and reporting*:

— *Part 1: Test reports*

Introduction

This International Standard will enhance the usability of biometric test data by providing them in a common and machine readable form. This International Standard is intended to provide

- documentary evidence that a product has been tested,
- a statement of authenticity of the test report,
- an ability to maintain of registry of products,
- a clear mechanism for maintaining product availability and certification status, and
- an ability for a relying system to depend on a biometric product used in a remote authentication context.

This International Standard is not intended to replace traditional biometric test reports. Indeed, because such texts are essential to the complete documentation of a test, they are viewed as parents of the machine readable content defined in ISO/IEC 29120 and are explicitly referenced in these reports.

Accordingly, the parts of this International Standard establish requirements for, and define formats for, signed test reports and biometric datasets as follows.

This part of ISO/IEC 29120 establishes machine readable records for documenting the output of a biometric test. This supports the documentary reporting requirements of some parts of ISO/IEC 19795. This part of ISO/IEC 29120 is primarily intended to support scenario and technology tests. Additionally, interoperability tests can be documented by a collection of ISO/IEC 29120-1 test reports (one for each tested combination of components). The International Standard also includes mechanism to protect the integrity of the test report. This assures a receiving system that the test information (date, laboratory, accreditation body, manner of testing, conformance, test size, accuracy) can be relied upon and used appropriately.

As the parts of ISO/IEC 19795 have been developed and testing standards have been published, there is an increasing reliance on the correct conduct of tests and their documented outputs. Although the ISO/IEC 19795 standards include extensive disclosure and reporting requirements, they do not establish definitive data formats for those pieces of information. Other data concerning the commissioning, accreditation, and conduct of the test can also be valuable to consumers of the test reports. In addition, this International Standard will benefit users of biometric tests via improved

- conformance to testing standards,
- reliability (via automation of relevant activities), and
- comparability of test results.

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 29120-1:2015

Information technology — Machine readable test data for biometric testing and reporting —

Part 1: Test reports

1 Scope

This part of ISO/IEC 29120 establishes

- machine readable records for documenting the output of a biometric test,
- formats for data that ISO/IEC 19795 tests are required to report, and
- an ASN.1 syntax for test reports.

This standard specifically does not

- require, prohibit, or otherwise specify, the format of biometric samples or templates used in a test,
- require, prohibit or otherwise specify, the encapsulation of biometric samples or templates used in a test, or
- regulate metrics for tests.

NOTE ISO/IEC 19795-1 establishes the reportable metrics.

2 Conformance

A test report shall be conformant to this part of ISO/IEC 29120 if it meets all normative requirements of this part of ISO/IEC 29120.

3 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 8601:2004, *Data elements and interchange formats — Information interchange — Representation of dates and times*

ISO/IEC 8825-1:2008, *Information technology — ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) — Part 1*

ISO/IEC 8825-4:2008, *Information technology — ASN.1 encoding rules: XML Encoding Rules (XER) — Part 4*

ISO/IEC 9594-2, *Information technology — Open Systems Interconnection — The Directory — Part 2: Models*

ISO/IEC 19785-3:2007, *Information technology — Common Biometric Exchange Formats Framework — Part 3: Patron format specifications*

ISO/IEC 19795-1, *Information technology — Biometric performance testing and reporting — Part 1: Principles and framework*

RFC 3852, *Cryptographic Message Syntax (CMS)*

RFC 5911, *New ASN.1 Modules for Cryptographic Message Syntax (CMS) and S/MIME*

4 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 19795-1 apply.

5 Symbols and abbreviated terms

For the purposes of this part of ISO/IEC 29120, the following abbreviations apply.

ASN	Abstract Syntax Notation
BDB	Biometric Data Block
CDF	Cumulative Distribution Function
CMC	Cumulative match characteristic
DET	Detection error tradeoff
FAR	False accept rate
FTA	Failure to acquire rate
FTE	Failure to enrol rate
FMR	False match rate
FNIR	False-negative identification error rate
FNMR	False non-match rate
FPIR	False-positive identification error rate
FRR	False reject rate
ROC	Receiver operating characteristic
IUT	Implementation under test

6 ASN.1 format

6.1 Encoding rules

The test reports specified in this part of the International Standard shall be encoded using the XML Encoding Rules (XER) [ISO/IEC 8825-4:2008] or the Basic Encoding Rules (BER) [ISO/IEC 8825-1:2008] of ASN.1.

6.2 ASN.1 object identifier for test report

```
MachineReadableBiometricTestingAndReportingTestReport {
    iso(1) standard(0) MRTDBTR(29120) testReport(1) module(1) rev(0)
}
```

6.3 BiometricTestReport type

```
BiometricTestReport ::= SEQUENCE {
    contentType    CONTENT-TYPE.&id({ContentTypeBiometricTestReport }),
    content        [0] EXPLICIT CONTENT-TYPE.&Type
```

```

    ({ContentTypeBiometricTestReport}{@contentType})
}

```

Type `BiometricTestReport` is composed of two components, `contentType` and `content`. The first component `contentType` is an object identifier, which indicates the type of content in the second component `content`. The value of `contentType` takes one of the following three values; `id-testReportTechnology`, `id-testReportScenario`, and `id-signedTestReport`. This is done by the following definition of `ContentTypeBiometricTestReport` and those of `testReportTechnology`, `testReportScenario`, and `signedDataBTR`.

```

ContentTypeBiometricTestReport CONTENT-TYPE ::= { testReportTechnology |
testReportScenario |
                                                    signedTestReport }

testReportTechnology CONTENT-TYPE ::= {
    TestReportTechnology
    IDENTIFIED BY id-testReportTechnology
}
testReportScenario CONTENT-TYPE ::= {
    TestReportScenario
    IDENTIFIED BY id-testReportScenario
}
signedTestReport CONTENT-TYPE ::= {
    SignedTestReport
    IDENTIFIED BY id-signedTestReport
}

```

Each of these content types corresponds to report of technology test, scenario test, and signed test report.

The object identifiers are defined as follows:

```

id-testReportTechnology OBJECT IDENTIFIER ::= {
    iso(1) standard(0) MRTDBTR(29120) testReport(1) contentType(2)
    testReportTechnology(1)
}
id-testReportScenario OBJECT IDENTIFIER ::= {
    iso(1) standard(0) MRTDBTR(29120) testReport(1) contentType(2)
    testReportScenario(2)
}
id-signedTestReport OBJECT IDENTIFIER ::= {
    iso(1) standard(0) MRTDBTR(29120) testReport(1) contentType(2) signedTestReport(3)
}

```

6.4 Data types for technology tests

6.4.1 Overview

Type `TestReportTechnology` is a type to express results of technology test. The first field `version` is the version of this test report format of type `MRTDBTRVersion`. The second field `targetInfo` is of type `ProductInformation` and gives information of the evaluated product. The third field `testReportInfo` gives the information about the test report of type `TestReportInformation`. The fourth part is a sequence `testReports` of type `TestReportTechnologyForOneCondition`. Each element of this sequence corresponds to a test result under a specific condition.

```

TestReportTechnology ::= SEQUENCE {
    version                MRTDBTRVersion        DEFAULT v0,
    targetInfo              ProductInformation,
    testReportInfo          TestReportInformation,
    testReports              SEQUENCE OF TestReportTechnologyForOneCondition
}
MRTDBTRVersion ::= INTEGER { v0(0) } ( v0, ... )

```

6.4.2 Product information

Type `ProductInformation` has six fields and gives information about the tested product.

```

ProductInformation ::= SEQUENCE {
    provider                Provider,

```

```

        nameProduct      NameProduct,
        description      VisibleString OPTIONAL,
        functionProduct   SEQUENCE OF Function,
        outputProduct     DataType OPTIONAL,
        modalityProduct   Modality
    }

```

6.4.2.1 Provider information

The first field `provider` is of type `Provider` and gives information about the provider of the tested biometric product.

```

Provider ::= SEQUENCE {
    nameProvider      Name,
    typeProvider      TypeProvider,
    roleProvider      RoleProvider,
    contactInformation VisibleString OPTIONAL
}

```

The first field `nameProvider` identifies the name of the provider. The type `Name` for this field is specified in ISO/IEC 9592-2:2005.

The second field `typeProvider` shows the type of the provider, and shall take a value chosen from the values of type `TypeProvider`: non-profit, university, corporation, individual, government.

```

TypeProvider ::= ENUMERATED {
    non-profit(1),
    university(2),
    corporation(3),
    individual(4),
    government(5)
}

```

The third field `roleProvider` shows the role of the provider, and shall take a value chosen from the values of type `RoleProvider`: manufacturer, reseller, integrator, other. `manufacturer` is for the role of the entity responsible for the design or creation of the component. `reseller` is for the role of the entity which packages or resells the component. `integrator` is for the role of the entity which may combine components into a single atomic component.

```

RoleProvider ::= ENUMERATED {
    manufacturer(1),
    reseller(2),
    integrator(3),
    other(4)
}

```

The fourth field `contactInformation`, which is optional, shows the contact information of the provider, such as the mail address of the provider, in `VisibleString`.

6.4.2.2 Other information in product information

The second field `nameProduct` in type `ProductInformation` is of `NameProduct` and gives basic information about the product.

```

NameProduct ::= SEQUENCE {
    modelName      Name,
    productCBEFF   Product OPTIONAL,
    version        VersionProduct,
    softwareVersion VersionProduct,
    firmwareVersion VersionProduct
}
VersionProduct ::= INTEGER { v0(0) } ( v0, ... )

```

The first field `modelName` in `NameProduct` is of type `Name` and identifies the product. The second field `productCBEFF` is an optional field of type `Product` imported from 19785-3 CBEFF Part 3. If the product is registered to a certain biometric organization, this field may be used to identify the product. The third, fourth, and fifth field `version`, `softwareVersion`, and `firmwareVersion` are all of type

VersionProduct and indicate the version of the product, the version of the software of the product, the version of the firmware of the product respectively.

The third field description in type ProductInformation gives a complete unique description of the component under the test in VisibleString. This field should be used to describe prototypes, experimental models, use of biometric modalities not listed in ISO/IEC 19785-3, or to give additional information about the biometric modality (e.g. for iris recognition in the visible spectrum).

The fourth field functionProduct in type ProductInformation expresses the function of the tested product with type Function.Type Function is specified as follows:

```
Function ::= ENUMERATED {
    acquisition(1),
    enrolment(2),
    verification(3),
    identification(4),
    ...
}
```

The fifth field outputProduct in type ProductInformation expresses the data type of the output of the tested product with type DataType. Type DataType consists of two fields, processedLevel and purpose. The former takes a value which corresponds to raw data, intermediate data, processed data, comparison score, or comparison decision. The latter takes a value which corresponds to biometric reference or biometric sample.

```
DataType ::= SEQUENCE {
    processedLevel          ProcessedLevel,
    purpose                 Purpose OPTIONAL
}
ProcessedLevel ::= ENUMERATED {
    raw-data(1),
    intermediate-data(2),
    processed-data(3),
    comparison-score(4),
    comparison-result(5),
    ...
}
Purpose ::= ENUMERATED {
    reference(1),
    sample(2)
}
```

The sixth field modalityProduct in type ProductInformation indicate the modality of biometric data which the tested product processes, with type Modality. Type modality consists of a pair of fields type and subtype. type is mandatory if processedLevel in outputProduct takes neither comparison-score nor comparison-result. The types BiometricType and BiometricSubtype are defined in ISO/IEC 19785-3:2007, Clause 6.2.

```
Modality ::= SEQUENCE {
    type          BiometricType,
    subtype       BiometricSubtype OPTIONAL
}
```

6.4.3 Information about test report

Type TestReportInformation has four fields and gives information about the test report.

```
TestReportInformation ::= SEQUENCE {
    testLabInformation          TestLabInformation,
    compliantStandard           StandardDescription,
    testReportIssuanceDate      Date,
    parentTestReport            ExternalDocument
}
```

The first field testLabInformation in type TestReportInformation identifies the test laboratory conducting the test, with type TestLabInformation. Type TestLabInformation consists of two fields: identificationTestLab of type IdentificationTestLab and accreditationStatus of type AccreditationStatus.

```
TestLabInformation ::= SEQUENCE {
    identificationTestLab IdentificationTestLab,
    accreditationStatus AccreditationStatus
}
```

Type IdentificationTestLab has five fields of type VisibleString : nameLab to show the name of the responsible laboratory, location to show location of the laboratory, optional testImplementor to show the employee or representative who executed the test, testReportSignatory to show the employee or representative assuring the integrity, correctness and completeness of the test, and contactInformation to show the contact information for enquiries concerning the test report.

```
IdentificationTestLab ::= SEQUENCE {
    nameLab VisibleString,
    location VisibleString,
    testImplementor VisibleString OPTIONAL,
    testReportSignatory VisibleString,
    contactInformation VisibleString
}
AccreditationStatus ::= SEQUENCE {
    accreditingBodies SEQUENCE OF AccreditingBody,
    scopeAccreditation ScopeAccreditation OPTIONAL
}
AccreditingBody ::= SEQUENCE {
    nameAccreditingBody VisibleString,
    identifierCertificate OBJECT IDENTIFIER,
    signatory OCTET STRING
}
```

The second field compliantStandard in type TestReportInformation indicates which testing standards were used for the test with type StandardDescription. Type StandardDescription has four fields: standardName in VisibleString to show the name of the standard such as Biometric Testing and Reporting – Principles and Framework, standardNumber in VisibleString to show the number of the standard such as 19795, standardPart in VisibleString to show the part number of the standard, and standardPublicationDate of type Date to show the publication date of the standard.

Type Date is expressed in VisibleString with fixed length of 8 of form YYYYMMDD, which conforms to ISO 8601.

```
StandardDescription ::= SEQUENCE {
    standardName VisibleString,
    standardNumber VisibleString,
    standardPart VisibleString,
    standardPublicationDate Date
}
Date ::= VisibleString
-- conforms to ISO 8601
-- length = 8
-- fixed
-- YYYYMMDD
```

The third field testReportIssuaranceDate in type TestReportInformation encodes the date on which the test report was signed by the test laboratory official with type Date.

The fourth field parentTestReport in type TestReportInformation gives the information about the non-machine readable, traditional test report for complete human-readable documentation of the test with type ExternalDocument. Type ExternalDocument consists of three mandatory fields and five optional fields. The first field link of type URI expresses the URL where the document can be referenced. The second field title of type VisibleString shows the title of the document. The third and optional field authors of type SEQUENCE OF VisibleString shows the author or the group of authors of the document. The fourth and optional field publisher of type VisibleString shows the publisher of the document. The fifth and optional field editor of type VisibleString shows the editor of the document. The sixth and optional field typeDocument of type TypeDocument shows the type of the document: article, technical report, in proceedings, abstract, book, in book, or collection. The seventh and optional field publicationDate of type Date shows the publication date

of the document. The eighth field **availability of type Availability** shows the availability of the document: public, restricted, unavailable, or superseded.

```
ExternalDocument ::= SEQUENCE {
    link                URI,
    title               VisibleString,
    authors             SEQUENCE OF VisibleString OPTIONAL,
    publisher           VisibleString OPTIONAL,
    editor              VisibleString OPTIONAL,
    typeDocument        TypeDocument OPTIONAL,
    publicationDate     Date OPTIONAL,
    availability         Availability
}
TypeDocument ::= ENUMERATED {
    article(1),
    technical-report(2),
    in-proceedings(3),
    abstract(4),
    book(5),
    in-book(6),
    collection(7)
}
Availability ::= ENUMERATED {
    public(1),
    restricted(2),
    unavailable(3),
    superseded(4)
}
```

6.4.4 Test report under a specific condition

6.4.4.1 Overview

Type **TestReportTechnologyForOneCondition** gives a set of information for a result of technology test under a given condition. **TestReportTechnologyForOneCondition** consists of four fields; **corpusInfo** of type **CorpusInformation**, **dateStarted** of type **Date**, **dateEnded** of type **Date**, and **testResult** of type **SEQUENCE OF TestResult**. The second and third are optional fields.

```
TestReportTechnologyForOneCondition ::= SEQUENCE {
    corpusInfo          CorpusInformation,
    dateStarted         Date OPTIONAL,
    dateEnded           Date OPTIONAL,
    testResult          SEQUENCE OF TestResult
}
```

6.4.4.2 Corpus information

Type **CorpusInformation** represents the information of the corpus which was used in the evaluation with two fields; **composition** of type **CorpusComposition** and **environInfo** of type **EnvironmentalInformation**.

```
CorpusInformation ::= SEQUENCE {
    composition         CorpusComposition,
    environInfo         EnvironmentalInformation
}
```

In type **CorpusComposition**, the corpus is identified with the first field identifier of type **OBJECT IDENTIFIER**. The second field **nameCorpus** of type **VisibleString** gives the name of the corpus. The third field **corpusStatistics** of type **CorpusStatistics** gives statistical information of the corpus.

```
CorpusComposition ::= SEQUENCE {
    identifier          OBJECT IDENTIFIER,
    nameCorpus          VisibleString,
    corpusStatistics    CorpusStatistics
}
```

Type `CorpusStatistics` consists of four fields. The first field `corpusBasicStatistics` of type `CorpusCrewBasicStatistics` gives the statistical information common to corpus and crew. The second field `numSamples` indicates the number of biometric samples in the test corpus. The mean number of samples per person can be obtained by dividing this number by the number of individuals `numIndividuals` in `corpusBasicStatistics`. The number of samples `numSamples` might be used in computation of uncertainties. The third and fourth fields, `samplesPerIndividualEnrol` and `samplesPerIndividualProbe`, are optional and indicate the number of enrolment samples per individual and the number of probe samples per individual respectively. Both are expressed with type `SamplesPerIndividual`.

```
CorpusStatistics ::= SEQUENCE {
    corpusBasicStatistics      CorpusCrewBasicStatistics,
    numSamples                 INTEGER,
    samplesPerIndividualEnrol   SamplesPerIndividual OPTIONAL,
    samplesPerIndividualProbe   SamplesPerIndividual OPTIONAL
}
```

Type `SamplesPerIndividual` is used to exhaustively tabulate a value for each member of the volunteer corpus. This type consists of four fields. The first field `numSubjects` indicates the number of subjects in the sample. The second and third fields, `mean` and `median`, are computed over all subjects. These two support applications that might not need data on the entire distribution expressed in `distrSubjSample`. The fourth field `distrSubjSample` is of type `DistributionIntegerInteger`, which is defined as `SEQUENCE OF ExpressionPointIntegerInteger`. Type `ExpressionPointIntegerInteger` consists of a pair of integers, `subjectId` and `numberOfSamples`. `numberOfSamples` expresses the number of samples for the `subjectId`. For example, 20 samples are given for subject ID 1, 30 samples for ID 2, 22 samples for ID 3, 16 samples for ID 4, 23 samples for ID 5, then `distrSubjSample` is ((1, 20), (2, 30), (3, 22), (4, 16), (5, 23)).

```
SamplesPerIndividual ::= SEQUENCE {
    numSubjects      INTEGER,
    mean             INTEGER,
    median           INTEGER,
    distrSubjSample   DistributionIntegerInteger
}
DistributionIntegerInteger ::= SEQUENCE OF ExpressionPointIntegerInteger
ExpressionPointIntegerInteger ::= SEQUENCE {
    subjectId        INTEGER,
    numberOfSamples   INTEGER
}
```

Type `CorpusCrewBasicStatistics` is used to express `corpusBasicStatistics` in `CorpusStatistics` and `testCrewInfo` in `TestReportScenarioForOneCondition` (see 6.4.1). This type consists of nine fields. The former five fields, `numIndividuals`, `numMales`, `numFemales`, `numIndividualsEnrol` and `numIndividualsVeriId`, are of type `INTEGER` and indicate the number of unique individuals in the test corpus/crew, that of male subjects, that of female subjects, that in the enrolment set, and that in the verification or identification set respectively. `numIndividuals` shall be equal to or greater than `numIndividualsEnrol` and `numIndividualsVeriId`. For identification, `numIndividualsVeriId` shall be the size of the population searched. The second and third fields are optional. The latter four fields `ageDistrMale`, `ageDistrFemale`, `elapsDistr`, and `visitsDayDistr`, are all optional and of type `InfoCumulativeDistribution`, and express the table of proportions of the males whose age in years is less or equal to X, the table of proportions of the females whose age in years is less or equal to X, the table of proportions of the subjects for whom the number of the days between the visits is less or equal to T, and the table of proportions of the samples collected on the day less or equal to n-th day, respectively.

```
CorpusCrewBasicStatistics ::= SEQUENCE {
    numIndividuals      INTEGER,
    numMales            INTEGER OPTIONAL,
    numFemales          INTEGER OPTIONAL,
    numIndividualsEnrol INTEGER,
    numIndividualsVeriId INTEGER,
    ageDistrMale        InfoCumulativeDistribution OPTIONAL,
    ageDistrFemale      InfoCumulativeDistribution OPTIONAL,
    elapsDistr          InfoCumulativeDistribution OPTIONAL,
}
```

```

visitsDayDistr
}
InfoCumulativeDistribution OPTIONAL

```

Type `InfoCumulativeDistribution` is used for tabulation and relevant information of the cumulative distribution function of a random variable. The first and second fields, `mean` and `median`, are computed over all `xValues` in `cumulativeDistribution`. These two fields support applications that do not need data on the entire cumulative distribution expressed in `cumulativeDistribution`. The third field `cumulativeDistribution` expresses the tabulation of the cumulative distribution with type `DistributionIntegerReal`, which is defined as `SEQUENCE OF ExpressionPointIntegerReal`. Each element of `DistributionIntegerReal` is a pair of `xValue` of type `INTEGER` and `yValue` of type `REAL`. An element of type `ExpressionPointIntegerReal` expresses that the proportion of the values which are less or equal to `xValue` is `yValue`. The elements shall appear in increasing order in `xValue`. For example, the expression of the following table in `DistributionIntegerReal` is `((0, 0), (1, 0), (2, 0.7), (3, 0.92), (4, 0.97), (5, 1))`.

xValue	yValue
0	0
1	0
2	0.7
3	0.92
4	0.97
5	1

```

InfoCumulativeDistribution ::= SEQUENCE {
    mean                INTEGER,
    median              INTEGER,
    cumulativeDistribution DistributionIntegerReal
}
DistributionIntegerReal ::= SEQUENCE OF ExpressionPointIntegerReal
ExpressionPointIntegerReal ::= SEQUENCE {
    xValue              INTEGER,
    yValue              REAL
}

```

To describe environment of the corpus collection, type `EnvironmentalInformation` is specified. The first field `exceptionalCondition` allows free text keywords indicating that the collection environment was adverse. The second field `celsiusTemp` represents the temperature expressed in Celsius in which the collection was done. The third field `dbNoise` represents the ambient noise expressed in dB in which the collection was done. The fourth and optional field `lightingInfo` allows free text in `VisibleString` to give the lighting information in which the collection was done.

```

EnvironmentalInformation ::= SEQUENCE {
    exceptionalCondition VisibleString,
    celsiusTemp          REAL OPTIONAL, -- temperature
    dbNoise              REAL OPTIONAL, -- ambient noise
    lightingInfo         VisibleString OPTIONAL
}

```

6.4.4.3 Test result under a specific condition

To express a test result for technology test, type `TestResult` is specified as follows. According to what is tested, i.e. enrolment, acquisition, matching in verification, or matching in identification, the component shall be chosen.

```

TestResult ::= CHOICE {
    testResultEnrol      TestResultEnrol, -- enrolment
    testResultAcquire    TestResultAcquire, -- acquisition
    testResultVerify     TestResultVerify, -- verification
    testResultIdentify   TestResultIdentify -- identification
}

```

6.4.4.3.1 Test result for enrolment

If the test is on enrolment, `testResultEnrol` of type `TestResultEnrol` shall be the component. Type `TestResultEnrol` consists of two fields, `failureToEnrolRate` and `durationEnrol`. The first field `failureToEnrolRate` expresses FTE, the fraction of enrolment samples not converted into a template. The second and optional field `durationEnrol` of type `StatisticInformationSet` gives statistical information on enrolment. This type gives a fundamental set of statistical information common to enrolment, acquisition, verification, and identification. The first field `unitTime` indicates the unit of time used in the third to the eighth field, millisecond or second. The second field is optional and indicates the number of measurements. Fields from the third to the eighth are optional and express the median, mean, minimum value, maximum value, standard deviation, and median absolute deviation of the value sets respectively.

```
TestResultEnrol ::= SEQUENCE {
    failureToEnrolRate      REAL,
    durationEnrol           StatisticInformationSet OPTIONAL
}
StatisticInformationSet ::= SEQUENCE {
    unitTime               UnitTime,
    numberOfMeasurements   INTEGER OPTIONAL,
    median                 REAL OPTIONAL,
    mean                   REAL OPTIONAL,
    minimum                REAL OPTIONAL,
    maximum                REAL OPTIONAL,
    stdDev                 REAL OPTIONAL,
    medAbsDev              REAL OPTIONAL
}
UnitTime ::= ENUMERATED {
    millisecond(1),
    second(2)
}
```

6.4.4.3.2 Test result for acquisition

If the test is on acquisition, `testResultAcquire` of type `TestResultAcquire` shall be the component in type `TestResult`. The type consists of `failureToAcquireRate` and optional `durationAcquire` of `StatisticInformationSet` type. The first field `failureToAcquireRate` expresses FTA, the fraction of acquisition samples not converted into a template.

```
TestResultAcquire ::= SEQUENCE {
    failureToAcquireRate    REAL,
    durationAcquire         StatisticInformationSet OPTIONAL
}
```

6.4.4.3.3 Test result for verification

For the test on matching in verification, type `TestResultVerify` is specified. This type consists of two fields; `resultMatchVerify` of type `ResultMatchVerify` and optional `durationVerify` of type `StatisticInformationSet`. The first three fields of type `ResultMatchVerify` are all of type `InfoDETCurve` and give information about three DET curves, `infoDETFNMRFMR` for the DET curve of FNMR and FMR, `infoDETFRRFAR` for the DET curve of FRR and FAR, and `infoDETGFRRGFAR` for the DET curve of GFR and GFAR. The fourth field of type `ResultMatchVerify` is the distribution of comparison scores `cmpScrDistr` of type `DistributionRealReal`.

```
TestResultVerify ::= SEQUENCE {
    resultMatchVerify      ResultMatchVerify,
    durationVerify         StatisticInformationSet OPTIONAL
}
ResultMatchVerify ::= SEQUENCE {
    infoDETFNMRFMR         InfoDETCurve, -- pair of error types shall be fnmr-fmr
    infoDETFRRFAR          InfoDETCurve, -- pair of error types shall be frr-far
    infoDETGFRRGFAR        InfoDETCurve, -- pair of error types shall be gfr-gfar
    cmpScrDistr            DistributionRealReal OPTIONAL
}
```

The first and second fields in type `InfoDETCurve` are the number of samples used in estimation of Type I estimate and that of Type II estimate. The third field `expressionDETCurve` approximates a

DET curve with type `InfoDETCurve`. `InfoDETCurve` represents a curve with an arbitrary number of points on the curve. Each point on the curve is expressed with `ExpressionPointDETCurve`, which is a triple of the threshold `threshold`, the Type I error rate value `typeIError`, and the Type II error rate value `typeIIError`. The sequence of points shall appear in increasing order in `typeIError`. If the threshold is unknown, `threshold` shall take the value -1. If the threshold is unavailable, `threshold` shall take the value 0.

```
InfoDETCurve ::= SEQUENCE {
    numOfSamplesEstTypeIError      INTEGER,
    numOfSamplesEstTypeIIError     INTEGER,
    expressionDETCurve             ExpressionDETCurve
}
ExpressionDETCurve ::= SEQUENCE OF ExpressionPointDETCurve
ExpressionPointDETCurve ::= SEQUENCE {
    threshold      REAL OPTIONAL, -- -1 for unavailable, -2 for unknown
    typeIError     REAL,
    typeIIError    REAL
}
```

Distribution of comparison score is expressed with type `DistributionRealReal` which is a sequence of `ExpressionPointRealReal`. Each element of `DistributionRealReal` is a pair of `xValue` of type `REAL` and `yValue` of type `REAL`. An element of type `ExpressionPointRealReal` expresses that the proportion of the values which are less or equal to `xValue` is `yValue`. The elements shall appear in increasing order in `xValue`.

```
DistributionRealReal ::= SEQUENCE OF ExpressionPointRealReal
ExpressionPointRealReal ::= SEQUENCE {
    xValue REAL,
    yValue REAL
}
```

6.4.4.3.4 Test result for identification

For the test on matching in identification, type `TestResultIdentify` is specified. This type consists of two fields; the result of closed-set identification `resultMatchClosedIdentify` of type `ResultMatchClosedIdentify` and the result of open-set identification `resultMatchOpenIdentify` of type `ResultMatchOpenIdentify` where the latter is optional.

```
TestResultIdentify ::= SEQUENCE {
    resultMatchClosedIdentify      ResultMatchClosedIdentify,
    resultMatchOpenIdentify       ResultMatchOpenIdentify OPTIONAL
}
```

NOTE Closed set metrics are mandatory because, as rank based statistics, they always be computed.

Type `ResultMatchClosedIdentify` consists of three fields; `cmcCurveClosed`, `srchExecDistr`, and `durationClosedIdentify`. `cmcCurveClosed` expresses the CMC curve of the test result with type `DistributionIntegerReal`. `srchExecDistr` expresses the histogram of number of searches executed in the closed-set identification. Type `ExpressionHistogram` represents a histogram with a sequence of `IntervalIntegerFrequency`. The first and second fields, `lowerLimit` and `upperLimit`, represent an interval, and the third field `frequency` represents the frequency on that interval. The elements in `ExpressionHistogram` shall appear in increasing order in `lowerLimit`. The last optional field `durationClosedIdentify` expresses the statistics of closed-set identification search duration with type `StatisticInformationSet`.

```
ResultMatchClosedIdentify ::= SEQUENCE {
    cmcCurveClosed      DistributionIntegerReal,
    srchExecDistr       ExpressionHistogram,
    durationClosedIdentify StatisticInformationSet OPTIONAL
}
ExpressionHistogram ::= SEQUENCE OF IntervalIntegerFrequency
IntervalIntegerFrequency ::= SEQUENCE {
    lowerLimit      INTEGER,
    upperLimit      INTEGER,
    frequency       INTEGER
}
```

Type `ResultMatchOpenIdentify` consists of five fields: the expression of the CMC curve `cmcCurveOpen`, the number of searches with enrolled mate `srchExecDistrEnrolled`, the number of

searches with no enrolled mate `srchExecDistrNoEnroled`, the information about the DET curve of FNIR and FPIR `infoDETCurveFNIRFPIR`, and the statistics of open-set identification search duration `durationOpenIdentify` where the fourth and fifth fields are optional. The types to express these fields are as follows and already defined.

```
ResultMatchOpenIdentify ::= SEQUENCE {
    cmcCurveOpen          DistributionIntegerReal,
    srchExecDistrEnroled   ExpressionHistogram,
    srchExecDistrNoEnroled ExpressionHistogram,
    infoDETCurveFNIRFPIR   InfoDETCurve OPTIONAL,
    -- pair of error types shall be fnir-fpir
    durationOpenIdentify   StatisticInformationSet OPTIONAL
}
```

6.5 Data types for scenario tests

6.5.1 Overview

Type `TestReportScenario` is the type to express results of scenario test. The first field `version` is the version of this test report format of type `MRTDBTRVersion`. The second field `targetInfos` is a sequence of type `ProductInformation` and gives information of set of the tested products. The third field `testReportInfo` gives the information about the test report of type `TestReportInformation`. The fourth field `testReports` is a sequence of type `TestReportScenarioForOneCondition`. Each element of this sequence corresponds to a test result under a specific condition. Types `testReportInfo` and `TestReportInformation` are already defined. For details, see [6.4.1](#) and [6.4.3](#).

```
TestReportScenario ::= SEQUENCE {
    version          MRTDBTRVersion          DEFAULT v0,
    targetInfos       SEQUENCE OF ProductInformation,
    testReportInfo    TestReportInformation,
    testReports       SEQUENCE OF TestReportScenarioForOneCondition
}
```

6.5.2 Test report under a specific condition

Type `TestReportScenarioForOneCondition` gives a set of information for a result of scenario test under a given condition. `TestReportScenarioForOneCondition` consists of six fields; `testCrewInfo` of type `TestCrewInformation`, `levelPolicyAssistance` of type `LevelPolicyAssistance`, `environInfo` of type `EnvironmentalInformation`, `dateStarted` of type `Date`, `dateEnded` of type `Date`, and `testResult` of a sequence of type `TestResult`. The fields of type `Date` are optional.

```
TestReportScenarioForOneCondition ::= SEQUENCE {
    testCrewInfo      TestCrewInformation,
    levelPolicyAssistance LevelPolicyAssistance,
    environInfo       EnvironmentalInformation,
    dateStarted       Date OPTIONAL,
    dateEnded         Date OPTIONAL,
    testResult        SEQUENCE OF TestResult
}
```

Type `TestCrewInformation` gives information on the test crew. The test crew is identified with the first field `identifier` of type `OBJECT IDENTIFIER`. The second field `location` in `VisibleString` is the information on the location where the scenario test is done. The third field `habituation` is expressed as a histogram of the past usage counts on the system with type `ExpressionHistogram`. The fourth field `testCrewStatistics` gathers statistical information of the test crew whose items are common to those for corpus. Type `CorpusCrewBasicStatistics` is defined in [6.4.4.2](#).

```
TestCrewInformation ::= SEQUENCE {
    identifier        OBJECT IDENTIFIER,
    location          VisibleString,
    habituation       ExpressionHistogram,
    testCrewStatistics CorpusCrewBasicStatistics
}
```

Type `LevelPolicyAssistance` describes the level of effort, decision policy, assistance provided, and instructional mode of the scenario test. This type has two fields, `levelEffortAndDecisionPolicy`

of type `LevelEffortAndDecisionPolicy` and optional `assistanceAndInstruction` of type `AssistanceAndInstruction`.

```
LevelPolicyAssistance ::= SEQUENCE {
    levelEffortAndDecisionPolicy LevelEffortAndDecisionPolicy,
    assistanceAndInstruction     AssistanceAndInstruction OPTIONAL
}
```

Type `LevelEffortAndDecisionPolicy` has two fields of type `LevelAndPolicy`, the enrolment policy `levelAndPolicyEnrol` and the comparison policy `levelAndPolicyCmp`. Type `LevelAndPolicy` consists of three fields; the minimum number of attempts, the maximum number of attempts, and the maximum duration permitted.

```
LevelEffortAndDecisionPolicy ::= SEQUENCE {
    levelAndPolicyEnrol LevelAndPolicy,
    levelAndPolicyCmp   LevelAndPolicy
}
LevelAndPolicy ::= SEQUENCE {
    minNumAttempt    INTEGER,
    maxNumAttempt    INTEGER,
    maxDurPermitted  REAL
}
```

Type `AssistanceAndInstruction` consists of three fields, `assistanceLocation`, `assistanceMode`, and `instructionMode`. The possible values for each field is specified as `AssistanceLocation`, `AssistanceMode`, and `InstructionMode` respectively.

```
AssistanceAndInstruction ::= SEQUENCE {
    assistanceLocation AssistanceLocation,
    assistanceMode      AssistanceMode,
    instructionMode      InstructionMode
}
AssistanceLocation ::= ENUMERATED {
    separate-from-transaction(1),
    interactively-with-transaction(2),
    after-failure(3)
}
AssistanceMode ::= ENUMERATED {
    physical(1),
    audio-only(2),
    audio-video(3),
    none(4)
}
InstructionMode ::= ENUMERATED {
    written-manual(1),
    poster(2),
    video(3),
    personal(4)
}
```

6.6 Data types for signed test reports

Type `SignedTestReport` is defined to express the signed test reports, test certificates in other words:

```
SignedTestReport ::= SEQUENCE {
    version                MRTDBTRVersion                DEFAULT v0,
    digestAlgorithms        DigestAlgorithmIdentifiers,
    encapContentInfo        EncapsulatedContentInfoSignedTR,
    certificates             [0] IMPLICIT CertificateSet OPTIONAL,
    crls                    [1] IMPLICIT RevocationInfoChoices OPTIONAL,
    signerInfos              SignerInfos
}
```

The `digestAlgorithms` component takes a value of type `DigestAlgorithmIdentifiers`, which is a collection of message digest algorithm identifiers. In this part of ISO/IEC 29120, the digest algorithm to be supported is not specified. For details of this type, see RFC 3852.

`encapContentInfo` contains the test result expressed in type `EncapsulatedContentInfoSignedTR`.

Type EncapsulatedContentInfoSignedTR is composed of two components, eContentTypeContentInfoSignedTR and eContentContentInfoSignedTR. The value of eContentTypeContentInfoSignedTR takes one of the following two values; id-testReportTechnology and id-testReportScenario. This is done by the following definition of ContentTypeContentInfoSignedTR and those of testReportTechnology and testReportScenario. eContentContentInfoSignedTR is the test report itself, carried as an octet string.

```
EncapsulatedContentInfoSignedTR ::= SEQUENCE {
    eContentTypeContentInfoSignedTR    CONTENT-TYPE.&id
        ({ContentTypeContentInfoSignedTR })),
    eContentContentInfoSignedTR        [0] EXPLICIT OCTET STRING
        (CONTAINING CONTENT-TYPE.&Type
        ({ContentTypeContentInfoSignedTR }{@contentType}))
}
ContentTypeContentInfoSignedTR CONTENT-TYPE ::= { testReportTechnology |
    testReportScenario }
```

certificates is a collection of certificates of type CertificateSet. It is intended that the set of certificates be sufficient to contain certification paths from a recognized “root” or “top-level certification authority” to all of the signers in the signerInfos field. For details of this type, see RFC 3852.

crls of type RevocationInfoChoices is a collection of revocation status information. It is intended that the collection contain information sufficient to determine whether the certificates in the certificates field are valid, but such correspondence is not necessary. For details of type RevocationInfoChoices, see RFC 3852.

signerInfos is a collection of per-signer information. For details of type SignerInfo type, RFC 3852.

Annex A (normative)

ASN.1 module for machine readable biometric test reports

A.1 Scope

This annex provides the complete ASN.1 module for this part of the ISO/IEC 29120. This Annex is the authoritative specification for binary encodings of these data elements, which shall be normally be specified in later parts of the ISO/IEC 29120 as ASN.1 Packed Encoding Rules (see X.691 | ISO/IEC 8825-2), unless there is a requirement to use security features that need the ASN.1 Basic Encoding Rules.

NOTE Software is available to convert between binary (both PER and Basic) and XML encodings using this ASN.1 specification. For XML encodings Annex (XSD) is authoritative, but the XSD Annex is conformant to ITU-T Rec. X.694|ISO/IEC 8825-5 and specifies the same XML encodings as the XSD Annex, if the ITU-T Rec. X.693 | 8825-4 encoding rules are applied.

A.2 ASN.1 module

```
MachineReadableBiometricTestingAndReportingTestReport {
    iso(1) standard(0) MRTDBTR(29120) testReport(1) module(1) rev(0)
}
DEFINITIONS AUTOMATIC TAGS ::= BEGIN
IMPORTS
-- ITU-T X.501 Open Systems Interconnection - The Directory: Models
    Name
    FROM InformationFramework {
        joint-iso-itu-t ds(5) module(1) informationFramework(1) 5}
-- RFC 3852/5911 Cryptographic Message Syntax
    DigestAlgorithmIdentifiers, CertificateSet,
    RevocationInfoChoices, SignerInfos, CONTENT-TYPE
    FROM CryptographicMessageSyntax-2009 {
        iso(1) member-body(2) us(840) rsadsi(113549)
        pkcs(1) pkcs-9(9) smime(16) modules(0) id-mod-cms-2004-02(41)}
-- ISO/IEC 19785 CBEFF Part 3
    BiometricType, BiometricSubtype, Product
    FROM CBEFF-DATA-ELEMENTS {
        iso standard 19785 modules(0) types-for-cbeff-data-elements(1) };
MRTDBTRVersion ::= INTEGER { v1(1) } ( v1, ... )
BiometricTestReport ::= SEQUENCE {
    contentType          CONTENT-TYPE.&id({ContentTypeBiometricTestReport }),
    content               [0] EXPLICIT CONTENT-TYPE.&Type
                        (({ContentTypeBiometricTestReport}){@contentType})
}
ContentTypeBiometricTestReport CONTENT-TYPE ::= { testReportTechnology |
testReportScenario|

signedTestReport }
TestReportTechnology ::= SEQUENCE {
    version              MRTDBTRVersion          DEFAULT v0,
    targetInfo           ProductInformation,
    testReportInfo       TestReportInformation,
    testReports          SEQUENCE OF TestReportTechnologyForOneCondition
}
ProductInformation ::= SEQUENCE {
    provider             Provider,
    nameProduct          NameProduct,
    description          VisibleString OPTIONAL,
    functionProduct      SEQUENCE OF Function,
    outputProduct        DataType OPTIONAL,
    modalityProduct      Modality
}
```

```

}
Provider ::= SEQUENCE {
    nameProvider          Name,
    typeProvider          TypeProvider,
    roleProvider          RoleProvider,
    contactInformation    VisibleString OPTIONAL
}
TypeProvider ::= ENUMERATED {
    non-profit(1),
    university(2),
    corporation(3),
    individual(4),
    government(5)
}
RoleProvider ::= ENUMERATED {
    manufacturer(1),
    reseller(2),
    integrator(3),
    other(4)
}
NameProduct ::= SEQUENCE {
    modelName             Name,
    productCBEFF          Product OPTIONAL,
    version               VersionProduct,
    softwareVersion       VersionProduct,
    firmwareVersion       VersionProduct
}
VersionProduct ::= INTEGER { v0(0) } ( v0, ... )
Function ::= ENUMERATED {
    acquisition(1),
    enrolment(2),
    verification(3),
    identification(4),
    ...
}
DataType ::= SEQUENCE {
    processedLevel        ProcessedLevel,
    purpose               Purpose OPTIONAL
}
ProcessedLevel ::= ENUMERATED {
    raw-data(1),
    intermediate-data(2),
    processed-data(3),
    comparison-score(4),
    comparison-result(5),
    ...
}
Purpose ::= ENUMERATED {
    reference(1),
    sample(2)
}
Modality ::= SEQUENCE {
    type                 BiometricType,
    subtype              BiometricSubtype OPTIONAL
}
TestReportInformation ::= SEQUENCE {
    testLabInformation    TestLabInformation,
    compliantStandard     StandardDescription,
    testReportIssuanceDate Date,
    parentTestReport      ExternalDocument
}
TestLabInformation ::= SEQUENCE {
    identificationTestLab IdentificationTestLab,
    accreditationStatus   AccreditationStatus
}
IdentificationTestLab ::= SEQUENCE {
    nameLab              VisibleString,
    location             VisibleString,
    testImplementor      VisibleString OPTIONAL,
    testReportSignatory  VisibleString,
    contactInformation    VisibleString
}

```

```

}
AccreditationStatus ::= SEQUENCE {
    accreditingBodies SEQUENCE OF AccreditingBody,
    scopeAccreditation ScopeAccreditation OPTIONAL
}
AccreditingBody ::= SEQUENCE {
    nameAccreditingBody VisibleString,
    identifierCertificate OBJECT IDENTIFIER,
    signatory OCTET STRING
}
StandardDescription ::= SEQUENCE {
    standardName VisibleString,
    standardNumber VisibleString,
    standardPart VisibleString,
    standardPublicationDate Date
}
Date ::= VisibleString
    -- conforms to ISO 8601
    -- length = 8
    -- fixed
    -- YYYYMMDD
ExternalDocument ::= SEQUENCE {
    link URI,
    title VisibleString,
    authors SEQUENCE OF VisibleString OPTIONAL,
    publisher VisibleString OPTIONAL,
    editor VisibleString OPTIONAL,
    typeDocument TypeDocument OPTIONAL,
    publicationDate Date OPTIONAL,
    availability Availability
}
URI ::= VisibleString (SIZE(1..MAX))
TypeDocument ::= ENUMERATED {
    article(1),
    technical-report(2),
    in-proceedings(3),
    abstract(4),
    book(5),
    in-book(6),
    collection(7)
}
Availability ::= ENUMERATED {
    public(1),
    restricted(2),
    unavailable(3),
    superseded(4)
}
TestReportTechnologyForOneCondition ::= SEQUENCE {
    corpusInfo CorpusInformation,
    dateStarted Date OPTIONAL,
    dateEnded Date OPTIONAL,
    testResult SEQUENCE OF TestResult
}
CorpusInformation ::= SEQUENCE {
    composition CorpusComposition,
    environInfo EnvironmentalInformation
}
CorpusComposition ::= SEQUENCE {
    identifier OBJECT IDENTIFIER,
    nameCorpus VisibleString,
    corpusStatistics CorpusStatistics
}
CorpusStatistics ::= SEQUENCE {
    corpusBasicStatistics CorpusCrewBasicStatistics,
    numSamples INTEGER,
    samplesPerIndividualEnrol SamplesPerIndividual OPTIONAL,
    samplesPerIndividualProbe SamplesPerIndividual OPTIONAL
}
CorpusCrewBasicStatistics ::= SEQUENCE {
    numIndividuals INTEGER,

```

```

numMales                INTEGER OPTIONAL,
numFemales               INTEGER OPTIONAL,
numIndividualsEnrol      INTEGER,
numIndividualsVeriId     INTEGER,
ageDistrMale             InfoCumulativeDistribution OPTIONAL,
ageDistrFemale           InfoCumulativeDistribution OPTIONAL,
elapsDistr               InfoCumulativeDistribution OPTIONAL,
visitsDayDistr           InfoCumulativeDistribution OPTIONAL
}
InfoCumulativeDistribution ::= SEQUENCE {
    mean                INTEGER,
    median              INTEGER,
    cumulativeDistribution DistributionIntegerReal
}
DistributionIntegerReal ::= SEQUENCE OF ExpressionPointIntegerReal
ExpressionPointIntegerReal ::= SEQUENCE {
    xValue              INTEGER,
    yValue              REAL
}
}
SamplesPerIndividual ::= SEQUENCE {
    numSubjects         INTEGER,
    mean                INTEGER,
    median              INTEGER,
    distrSubjSample     DistributionIntegerInteger
}
DistributionIntegerInteger ::= SEQUENCE OF ExpressionPointIntegerInteger
ExpressionPointIntegerInteger ::= SEQUENCE {
    subjectId           INTEGER,
    numberOfSamples     INTEGER
}
}
EnvironmentalInformation ::= SEQUENCE {
    exceptionalCondition VisibleString,
    celsiusTemp          REAL OPTIONAL, -- temperature
    dBNoise              REAL OPTIONAL, -- ambient noise
    lightingInfo         VisibleString OPTIONAL
}
}
TestResult ::= CHOICE {
    testResultEnrol      TestResultEnrol, -- enrolment
    testResultAcquire    TestResultAcquire, -- acquisition
    testResultVerify     TestResultVerify, -- verification
    testResultIdentify   TestResultIdentify -- identification
}
}
TestResultEnrol ::= SEQUENCE {
    failureToEnrolRate   REAL,
    durationEnrol        StatisticInformationSet OPTIONAL
}
}
StatisticInformationSet ::= SEQUENCE {
    unitTime             UnitTime,
    numberOfMeasurements INTEGER OPTIONAL,
    median               REAL OPTIONAL,
    mean                 REAL OPTIONAL,
    minimum              REAL OPTIONAL,
    maximum              REAL OPTIONAL,
    stdDev               REAL OPTIONAL,
    medAbsDev            REAL OPTIONAL
}
}
UnitTime ::= ENUMERATED {
    millisecond(1),
    second(2)
}
}
TestResultAcquire ::= SEQUENCE {
    failureToAcquireRate REAL,
    durationAcquire      StatisticInformationSet OPTIONAL
}
}
TestResultVerify ::= SEQUENCE {
    resultMatchVerify    ResultMatchVerify,
    durationVerify       StatisticInformationSet OPTIONAL
}
}
ResultMatchVerify ::= SEQUENCE {
    InfoDETFNMRfMR      InfoDETCurve, -- pair of error types shall be fnmr-fmr
    infoDETFRRfAR       InfoDETCurve, -- pair of error types shall be frr-far

```

```

        infoDETFERRGFAR      InfoDETCurve, -- pair of error types shall be gfrr-gfar
        cmpScrDistr          DistributionRealReal OPTIONAL
    }
    InfoDETCurve ::= SEQUENCE {
        numOfSamplesEstTypeIError INTEGER,
        numOfSamplesEstTypeIIError INTEGER,
        expressionDETCurve      ExpressionDETCurve
    }
    ExpressionDETCurve ::= SEQUENCE OF ExpressionPointDETCurve
    ExpressionPointDETCurve ::= SEQUENCE {
        threshold              REAL OPTIONAL, -- 0 for unavailable, -1 for unknown
        typeIError              REAL,
        typeIIError             REAL
    }
    DistributionRealReal ::= SEQUENCE OF ExpressionPointRealReal
    ExpressionPointRealReal ::= SEQUENCE {
        xValue REAL,
        yValue REAL
    }
    TestResultIdentify ::= SEQUENCE {
        resultMatchClosedIdentify ResultMatchClosedIdentify,
        resultMatchOpenIdentify   ResultMatchOpenIdentify OPTIONAL
    }
    ResultMatchClosedIdentify ::= SEQUENCE {
        cmcCurveClosed           DistributionIntegerReal,
        srchExecDistr             ExpressionHistogram,
        durationClosedIdentify    StatisticInformationSet OPTIONAL
    }
    ExpressionHistogram ::= SEQUENCE OF IntervalIntegerFrequency
    IntervalIntegerFrequency ::= SEQUENCE {
        lowerLimit               INTEGER,
        upperLimit               INTEGER,
        frequency                INTEGER
    }
    ResultMatchOpenIdentify ::= SEQUENCE {
        cmcCurveOpen             DistributionIntegerReal,
        srchExecDistrEnroled      ExpressionHistogram,
        srchExecDistrNoEnroled    ExpressionHistogram,
        infoDETCurveFNIRFPIR      InfoDETCurve OPTIONAL,
        -- pair of error types shall be fnir-fpir
        durationOpenIdentify      StatisticInformationSet OPTIONAL
    }
    TestReportScenario ::= SEQUENCE {
        version                   MRTDBTRVersion          DEFAULT v0,
        targetInfos               SEQUENCE OF ProductInformation,
        testReportInfo            TestReportInformation,
        testReports               SEQUENCE OF TestReportScenarioForOneCondition
    }
    TestReportScenarioForOneCondition ::= SEQUENCE {
        testCrewInfo              TestCrewInformation,
        levelPolicyAssistance      LevelPolicyAssistance,
        environInfo               EnvironmentalInformation,
        dateStarted               Date OPTIONAL,
        dateEnded                 Date OPTIONAL,
        testResult                 SEQUENCE OF TestResult
    }
    TestCrewInformation ::= SEQUENCE {
        identifier                OBJECT IDENTIFIER,
        location                  VisibleString,
        habituation                ExpressionHistogram,
        testCrewStatistics         CorpusCrewBasicStatistics
    }
    LevelPolicyAssistance ::= SEQUENCE {
        levelEffortAndDecisionPolicy LevelEffortAndDecisionPolicy,
        assistanceAndInstruction     AssistanceAndInstruction OPTIONAL
    }
    LevelEffortAndDecisionPolicy ::= SEQUENCE {
        levelAndPolicyEnrol LevelAndPolicy,
        levelAndPolicyCmp    LevelAndPolicy
    }
    LevelAndPolicy ::= SEQUENCE {

```

```

        minNumAttempt      INTEGER,
        maxNumAttempt      INTEGER,
        maxDurPermitted    REAL
    }
    AssistanceAndInstruction ::= SEQUENCE {
        assistanceLocation    AssistanceLocation,
        assistanceMode        AssistanceMode,
        instructionMode        InstructionMode
    }
    AssistanceLocation ::= ENUMERATED {
        separate-from-transaction(1),
        interactively-with-transaction(2),
        after-failure(3)
    }
    AssistanceMode ::= ENUMERATED {
        physical(1),
        audio-only(2),
        audio-video(3),
        none(4)
    }
    InstructionMode ::= ENUMERATED {
        written-manual(1),
        poster(2),
        video(3),
        personal(4)
    }

    SignedTestReport ::= SEQUENCE {
        version                MRTDBTRVersion          DEFAULT v0,
        digestAlgorithms        DigestAlgorithmIdentifiers,
        encapContentInfo        EncapsulatedContentInfoSignedTR,
        certificates            [0] IMPLICIT CertificateSet OPTIONAL,
        crls                    [1] IMPLICIT RevocationInfoChoices OPTIONAL,
        signerInfos              SignerInfos
    }
    EncapsulatedContentInfoSignedTR ::= SEQUENCE {
        eContentTypeContentInfoSignedTR    CONTENT-TYPE.&id
            ({ContentTypeContentInfoSignedTR }),
        eContentContentInfoSignedTR        [0] EXPLICIT OCTET STRING
            ( CONTAINING CONTENT-TYPE.&type
              ({ContentTypeContentInfoSignedTR }{@contentType}))
    }
    ContentTypeContentInfoSignedTR CONTENT-TYPE ::= { testReportTechnology |
    testReportScenario }

    -- contentType object identifiers
    id-testReportTechnology OBJECT IDENTIFIER ::= {
        iso(1) standard(0) MRTDBTR(29120) testReport(1) contentType(2)
    }
    testReportTechnology(1)
    }
    id-testReportScenario OBJECT IDENTIFIER ::= {
        iso(1) standard(0) MRTDBTR(29120) testReport(1) contentType(2)
        testReportScenario(2)
    }
    testReportScenario(2)
    }
    id-signedTestReport OBJECT IDENTIFIER ::= {
        iso(1) standard(0) MRTDBTR(29120) testReport(1) contentType(2) signedTestReport(3)
    }
    }

    -- ContentType objects
    testReportTechnology CONTENT-TYPE ::= {
        TestReportTechnology
        IDENTIFIED BY id-testReportTechnology
    }
    testReportScenario CONTENT-TYPE ::= {
        TestReportScenario
        IDENTIFIED BY id-testReportScenario
    }
    }
    signedTestReport CONTENT-TYPE ::= {
        SignedTestReport
        IDENTIFIED BY id-signedTestReport
    }

```

}

END -- BIOMETRIC-TESTING-REPORTING-TEST-REPORT

STANDARDSISO.COM : Click to view the full PDF of ISO/IEC 29120-1:2015

Annex B (informative)

Common elements

B.1 Purpose

The following subsections describe the content and markup of data elements included in the test reports. This subclause describes data elements common to two or more of the test types specified in [C.2](#) and [C.3](#). This Annex is labeled as informative because the ultimate set of normative requirements are those given in [Clause 6](#).

B.2 Notation

The requirements established for test report data elements in this part of ISO/IEC 29120 are accompanied by labels “M” for mandatory, and “O” for optional. A test report shall include data elements labelled “M”. A report should include elements labelled “O”. Any optional elements shall be encoded according to the requirements given in this part of ISO/IEC 29120.

B.3 Biometric component provider

This data element identifies the manufacturer or supplier of the component under test. This element shall conform to the requirements of [Table B.1](#).

Table B.1 — Data elements for describing the provider of a biometric component

Elements		Status	Contents
Pro- vider	Name	M	Name of the provider
	Non-profit University Corporation Individual Government	M	Type of provider
	Manufacturer Reseller Integrator Other	M	Role of provider. The manufacturer is the entity responsible for the design or creation of the component. A reseller packages or re-sells the component. An integrator may combine components into a single atomic component.
	Contact information	O	An email address, or address, or phone number

NOTE The Table establishes requirements on reporting. However, the specific encoding of the data into a machine readable test report is defined by the normative Annexes.

B.4 Biometric component

An evaluation is conducted on a target. A target consists of one or more biometric components. Each component shall be identified according to the requirements of [Table B.2](#).

NOTE Some tests may be run on composite systems in which bits were delivered or updated on different days. This model of a target as an assembly of components allows for revision of parts of the system and on-the-fly update of the implementation under test.

Table B.2 — Data elements for describing each component of the evaluation

Elements		Sta- tus	Num entries		Contents
			Min	Max	
Name	Provider	M	1	1	Vendor (manufacturer/provider). This element shall conform to the requirements of Table B.1 .
	Model	M	1	∞	The model and version number shall be provided for commercial off-the-shelf products.
	Version	M	1	1	
	Software version	M	1	1	The value can be stated as being unknown, unspecified or unused (it doesn't have firmware, for example).
	Firmware version	M	1	1	
Instance	Date acquired	M	1	1	Date component was acquired by test laboratory.
	Unique model identifier	O	0	1	An identifier unique to this instance of this component
	Parameters	O	0	∞	Configurable hardware and software parameters. A list of name-value pairs. The name describe the parameter, the value gives a numeric or other value. Both fields should be free text. For components with no such parameters use “None”.
Type	CBEFF_BDB_product_type	O	0	1	Identifier defined in ISO/IEC 19785-3 CBEFF data element
Description	Arbitrary text	O	0	1	In cases where the component cannot be completely identified using model number fields, for example when the component is a prototype or an experimental version, this field shall be populated to give a complete and adequate description of the component under test.
Function	Acquisition processing storage enrolment verification identifica- tion comparison	M	1	∞	One or more of these may apply, because some components are multifunctional.
Type of output	None Other Biometric template Biometric sample Comparison score Accept/Reject decision Candidate list without comparison scores Candidate list with comparison scores Quality score	O	0	∞	The type of output of the component.
Modality	Type	M	1	∞	The particular biometric modality, per ISO/IEC 19785-3:2007, Clause 6. EXAMPLE: Face or iris.
	Subtype	O	0	∞	The particular biometric part of the modality, per ISO/IEC 19795-3:2007, Clause 6. EXAMPLE: Index finger, code:

B.5 Test laboratory

Information on the test laboratory conducting the test shall be recorded using the data elements of [Table B.3](#).

Table B.3 — Data elements identifying the test laboratory

Elements		Sta- tus	Num. entries		Contents
			Min	Max	
Identifica- tion	Lab Name	M	1	1	The name of the responsible laboratory.
	Location	M	1	∞	The location of the responsible laboratory.
	Test implementer	O	0	∞	The employee or representative who executed the test
	Test report signa- tory	M	1	1	The employee or representative assuring the integrity, correctness and completeness of the test.
	Contact informa- tion	M	1	∞	Contact information for enquiries concerning the test report
Accredita- tion Status	Name of accredi- ting body	M	1	∞	List of bodies accrediting the lab. If no accreditation is claimed this field shall be populated with an “accredita- tion not claimed” entry.
	Identifier for accreditation cer- tificate	M	1	∞	Identifier of the accreditation result
	Scope of accredi- tation	O	0	∞	EXAMPLE Claim of accreditation to do ISO/IEC 19795- 5:2011 testing
	Accreditor’s signa- tory	M	1	∞	Location, contact point, pointer, URI, or other reference to the accreditation certificate of a test laboratory.

NOTE ISO/IEC 29120 as a standard for formatting test data does not establish requirements on the conduct of a test, nor on the qualifications of a test laboratory. Particularly the presence of the accreditation field does not imply any need for accreditation of test labs – it merely supports identification of any relevant accreditations.

B.6 Standards compliance

All citations of standards shall conform to the requirements of [Table B.4](#). It allows a test laboratory to indicate which testing standards were used for the test. If this data element is used it indicates the laboratory is claiming conformance to the standard.

EXAMPLE A test might claim conformance to ISO/IEC 19795-4:2008 for the execution of the test, and to ISO/IEC 29109-2:2012 for the reporting of the test data.

NOTE If a standard claims conformance to more than one standard, then a this data element may be repeated in a suitable encapsulating structure.

Table B.4 — Data elements identifying a standard

Elements		Sta- tus	Contents
Standard	Name	M	EXAMPLE: Biometric Testing and Reporting - Principles and Framework
	Standard identifier	M	This field shall give a complete identification of the standard including organization, number, part (if any) and date. It should also include any relevant published amendments. EXAMPLES: ISO/IEC 19795-1:2005 RFC 2119 ISO/IEC 19784-1:2006/Amdt, 2007
	Supplemental information	O	Further information refining the use of the standard. This might include profile information, or a description of which parts of the standard are applicable. The content and format is not regulated by this part of ISO/IEC 29120.

B.7 Dates

All dates shall be conformant instances of ISO 8601:2004 Data elements and interchange formats -- Information interchange -- Representation of dates and times.

B.8 External documentation

The machine readable test report defined in this part of ISO/IEC 29120 is not intended as complete documentation of a test. Instead a larger traditional written test report may exist. In addition a formal written test plan or other document may exist. When such documents are referenced in a machine readable test report they should conform to the requirements of [Table B.5](#).

Table B.5 — Data elements of externally linked documents

Elements		Sta- tus	Contents
External Document	Link	M	A URI or webpage or
	Title	M	Performance of compact standard plantar images
	Authors	O	
	Publisher	O	
	Edition	O	
	Type. One of "Article", "Technical Report", "In Proceedings", "Abstract", "Book", "In book", "Collection"	O	
	Publication date YYYY-MM-DD	O	EXAMPLE: 2010-02-13
	Availability	M	Public Restricted Unavailable Superseded

B.9 Summary statistics for univariate data

If a test measures scalar quantities and these are summarized in the machine readable test report, these shall be named and encoded according to the requirements of [Table B.6](#). If the variable is a random variable, (e.g. the mean age of a volunteer crew) then the number of measurements over which the

random variable is estimated shall be reported i.e. status becomes M. If the variable is not a random variable (e.g. the size of the test crew), the number of measurements shall be set to 1.

Table B.6 — Data elements for summary statistics

Elements		Status	Contents
Statistic	Units	M	EXAMPLE: Milliseconds
	Number of measurements	M	This field is optional for variables that are not random variables. EXAMPLE 200.
	Median	O	A numeric value
	Mean	O	A numeric value
	Minimum	O	A numeric value
	Maximum	O	A numeric value
	Standard deviation	O	A numeric value
	Median absolute deviation	O	A numeric value

NOTE Biometric testing and reporting standards such as ISO/IEC 19795 may require specific variables and statistics to be reported.

B.10 Subject-specific data

This data element of [Table B.7](#) shall be used to exhaustively tabulate a value for each member of the volunteer corpus.

Table B.7 — Data elements for a subject-specific data

Elements			Status	Contents
Subject-specific data	Number of subjects		M	EXAMPLE: 200
	Name		M	Name of the variable e.g. iris diameter (in millimeters)
	Mean		M	These values are computed over all subjects. They support applications that might not need data on the entire crew.
	Median		M	
	Complete array of (ID, value) pairs	Subject ID	M	EXAMPLE: 12345
		Value	M	EXAMPLE: 11.2

B.11 Cumulative distribution function

The data element of [Table B.8](#) shall be used for tabulation of the cumulative distribution function of a random variable. That is, an entry in the table shall give the proportion of measurements for which the observed value is less or equal to the given value.

The elements shall appear in increasing order. For any given pair of elements X_k and X_{k+1} , the tabulated value $F(X_k)$ shall be less than or equal to $F(X_{k+1})$ for all indices, k . The range of X values shall be such that $F(X_1) = 0$ and $F(X_N) = 1$.

Table B.8 — Data elements for a cumulative distribution function

Elements		Sta- tus	Contents
Summary statistics	Mean value	M	These values support applications that do not need the entire CDF.
	Median value	M	
	Variance	O	
CDF	X	M	A list of pairs of x and F(x).
	FX	M	

NOTE The name of the variable is given in the enclosing data structure which embeds the [Table B.8](#) data.

EXAMPLE The element encodes data like the following. A real instance of this format would usually have many more entries.

X	F(X)
0	0
1	0
2	0.7
3	0.92
4	0.97
5	1

B.12 Detection error tradeoff characteristic

This element is a tabulation of measurements of the Type I and Type II error rates as functions of a operating threshold. This element shall conform to the requirements of [Table B.9](#). The Type I and II error rate names are paired as follows:

- If Type I error rate is “FMR” the Type II error rate shall be “FNMR”,
- If Type I error rate is “FAR” the Type II error rate shall be “FRR”,
- If Type I error rate is “GFAR” the Type II error rate shall be “GFRR”, and
- If Type I error rate is “FPIR” the Type II error rate shall be “FNIR”.

Table B.9 — Data elements for DET characteristic

Elements		Status	Contents
DET	Name of Type I error	M	EXAMPLE FAR
	Number of comparisons or transactions used in estimation of Type I estimate	M	
	Name of Type II error	M	EXAMPLE FRR
	Number of comparisons or transactions used in estimation of Type II estimate	M	
	DET points	T	Three values (T, E1, E2) where T is the threshold,
		E1	
		E2	E1 is the Type I error rate value, E2 is the Type II error rate value. T may be stated to be “unknown” or “unavailable”.

EXAMPLE The element encodes data like the following. A real instance of this format would usually have many more entries.

T = Threshold	E1 =FMR	E2 = FNMR
0.32	0.000001	0.004
0.33	0.000008	0.003
0.34	0.000064	0.002

In this example, the “Name of Type I variable” would be “FMR”.

Annex C (informative)

Test reports

C.1 Purpose

This Annex is included as informative text to provide implementers with an overview of the content of technology and scenario test reports. The clause is labeled as informative because any given test report instance is required to conform to the mandatory grammar specifications expressed using ASN.1. These are given in the [Clause 6](#) and [Annex A](#) both of which are normative.

C.2 Data elements for technology tests

A technology test report shall record the mandatory elements identified in [Table C.1](#). All items shall be formatted such that the test report is a conformant instance of the ASN.1 specification whose schema appears in normative [Annex A](#).

The test report defined by [Table C.1](#) is not intended as complete documentation of a test. It is likely to be an extract of a larger traditional written test report. A link to this parent is provided in the [Table C.1](#) record. The parent report will almost certainly not be machine readable but will serve as the reference document for users needing detailed complete information beyond that encoded in the machine readable extract.

NOTE This part of ISO/IEC 29120 does not usually regulate which measurements must be made, and how. Requirements for that are established by the various parts of ISO/IEC 19795.

Table C.1 — Data elements for technology test reports

Item	Elements	Nested Elements	Status	Contents
Test laboratories	Primary Laboratory	Table B.3	M	The laboratory conducting the test, or coordinating the test.
	Secondary Laboratories	Table B.3	O	Additional laboratories involved in the execution of the test, e.g. in case of inter-laboratory testing, shall be identified also according to the requirements of Table B.3
Target of the evaluation	Target	List of components conformant to Table B.2	M	This element encodes the target of the evaluation, i.e. the implementation under test.
External context	Name	Name of a parent test program or campaign	O	Indicate whether this test report is part of a larger set, e.g. as a test of one set of products in a larger multi-product interoperability test such as ILO. ^[3]
Test report issuance date	Report Date	Date, clause 6.7.	M	This field encodes the date on which the test report was signed by the test laboratory official.
Parent test report	Non-machine readable test report	External document, clause 6.8	O	Non-machine readable, traditional test report, for complete human-readable documentation of the test.
Corpus	Name		M	EXAMPLE "FVC 2004 Database 1"
	Object identifier		O	Owner identified, or identifier independently registered (e.g. under a SC37 standing document)