



Methods for Testing and Specification (MTS); The Test Description Language (TDL); Part 1: Abstract Syntax and Associated Semantics

*Full standard available at:
<https://standards.iteh.ai/catalog/standards/sic/9900/etsi-es-203-119-1-v1-3-1-2016-09>*

Reference

RES/MTS-203119-1v1.3.1

Keywords

language, MBT, methodology, testing, TSS&TP,
TTCN-3, UML

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:

<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at

<https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:

<https://portal.etsi.org/People/CommitteeSupportStaff.aspx>

Copyright Notification

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2016.

All rights reserved.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are Trade Marks of ETSI registered for the benefit of its Members.

3GPP™ and **LTE™** are Trade Marks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

GSM® and the GSM logo are Trade Marks registered and owned by the GSM Association.

Contents

Intellectual Property Rights	8
Foreword.....	8
Modal verbs terminology.....	8
1 Scope	9
2 References	9
2.1 Normative references	9
2.2 Informative references.....	10
3 Definitions and abbreviations.....	10
3.1 Definitions.....	10
3.2 Abbreviations	11
4 Basic Principles	11
4.1 What is TDL?	11
4.2 Design Considerations.....	12
4.3 Principal Design Approach.....	13
4.4 Document Structure.....	14
4.5 Notational Conventions	15
4.6 OCL Constraints Requirements.....	15
4.7 Conformance	15
5 Foundation.....	15
5.1 Overview	15
5.2 Abstract Syntax and Classifier Description.....	16
5.2.1 Element	16
5.2.2 NamedElement	17
5.2.3 PackageableElement	17
5.2.4 Package.....	18
5.2.5 ElementImport	18
5.2.6 Comment	19
5.2.7 Annotation	20
5.2.8 AnnotationType	20
5.2.9 TestObjective.....	21
6 Data	21
6.1 Overview	21
6.2 Data Definition - Abstract Syntax and Classifier Description.....	22
6.2.1 DataResourceMapping.....	22
6.2.2 MappableDataElement.....	22
6.2.3 DataElementMapping.....	23
6.2.4 ParameterMapping.....	24
6.2.5 DataType	25
6.2.6 DataInstance	25
6.2.7 SimpleDataType	26
6.2.8 SimpleDataInstance	26
6.2.9 StructuredDataType	27
6.2.10 Member.....	27
6.2.11 StructuredDataInstance.....	28
6.2.12 MemberAssignment.....	29
6.2.13 Parameter	30
6.2.14 FormalParameter.....	30
6.2.15 Variable	31
6.2.16 Action	31
6.2.17 Function	32
6.2.18 UnassignedMemberTreatment.....	32
6.3 Data Use - Abstract Syntax and Classifier Description.....	33
6.3.1 DataUse	33

6.3.2	ParameterBinding	34
6.3.3	StaticDataUse	35
6.3.4	DataInstanceUse	35
6.3.5	SpecialValueUse	36
6.3.6	AnyValue	37
6.3.7	AnyValueOrOmit	37
6.3.8	OmitValue	38
6.3.9	DynamicDataUse	38
6.3.10	FunctionCall	39
6.3.11	FormalParameterUse	39
6.3.12	VariableUse	39
7	Time	40
7.1	Overview	40
7.2	Abstract Syntax and Classifier Description	40
7.2.1	Time	40
7.2.2	TimeLabel	41
7.2.3	TimeLabelUse	42
7.2.4	TimeConstraint	42
7.2.5	TimeOperation	44
7.2.6	Wait	45
7.2.7	Quiescence	45
7.2.8	Timer	46
7.2.9	TimerOperation	46
7.2.10	TimerStart	47
7.2.11	TimerStop	47
7.2.12	TimeOut	48
8	Test Configuration	48
8.1	Overview	48
8.2	Abstract Syntax and Classifier Description	49
8.2.1	GateType	49
8.2.2	GateInstance	49
8.2.3	ComponentType	50
8.2.4	ComponentInstance	51
8.2.5	ComponentInstanceRole	51
8.2.6	GateReference	51
8.2.7	Connection	52
8.2.8	TestConfiguration	53
9	Test Behaviour	54
9.1	Overview	54
9.2	Test Description - Abstract Syntax and Classifier Description	55
9.2.1	TestDescription	55
9.2.2	BehaviourDescription	56
9.3	Combined Behaviour - Abstract Syntax and Classifier Description	57
9.3.1	Behaviour	57
9.3.2	Block	58
9.3.3	CombinedBehaviour	58
9.3.4	SingleCombinedBehaviour	59
9.3.5	CompoundBehaviour	59
9.3.6	BoundedLoopBehaviour	59
9.3.7	UnboundedLoopBehaviour	60
9.3.8	MultipleCombinedBehaviour	60
9.3.9	AlternativeBehaviour	61
9.3.10	ConditionalBehaviour	61
9.3.11	ParallelBehaviour	62
9.3.12	ExceptionalBehaviour	63
9.3.13	DefaultBehaviour	64
9.3.14	InterruptBehaviour	64
9.3.15	PeriodicBehaviour	65
9.4	Atomic Behaviour - Abstract Syntax and Classifier Description	66
9.4.1	AtomicBehaviour	66

9.4.2	Break.....	67
9.4.3	Stop.....	67
9.4.4	VerdictAssignment.....	67
9.4.5	Assertion.....	68
9.4.6	Interaction.....	69
9.4.7	Target.....	72
9.4.8	TestDescriptionReference.....	73
9.4.9	ComponentInstanceBinding.....	76
9.4.10	ActionBehaviour.....	77
9.4.11	ActionReference.....	78
9.4.12	InlineAction.....	78
9.4.13	Assignment.....	78
10	Predefined TDL Model Instances.....	79
10.1	Overview.....	79
10.2	Predefined Instances of the 'SimpleDataType' Element.....	79
10.2.1	Boolean.....	79
10.2.2	Verdict.....	79
10.2.3	TimeLabelType.....	80
10.3	Predefined Instances of 'SimpleDataInstance' Element.....	80
10.3.1	true.....	80
10.3.2	false.....	80
10.3.3	pass.....	80
10.3.4	fail.....	80
10.3.5	inconclusive.....	80
10.4	Predefined Instances of 'Time' Element.....	80
10.4.1	Second.....	80
10.5	Predefined Instances of the 'Function' Element.....	80
10.5.1	Overview.....	80
10.5.2	Functions of Return Type 'Boolean'.....	81
10.5.3	Functions of Return Type 'TimeLabelType'.....	81
10.5.4	Functions of Return Type of Instance of 'Time'.....	81
Annex A (informative):	Technical Representation of the TDL Meta-Model.....	82
Annex B (informative):	Examples of a TDL Concrete Syntax.....	83
B.1	Introduction.....	83
B.2	A 3GPP Conformance Example in Textual Syntax.....	83
B.3	An IMS Interoperability Example in Textual Syntax.....	85
B.4	An Example Demonstrating TDL Data Concepts.....	87
B.5	TDL Textual Syntax Reference.....	89
B.5.1	Conventions for the TDLan Syntax Definition.....	89
B.5.2	TDL Textual Syntax EBNF Production Rules.....	89
Annex C (normative):	TDL - UML Mapping.....	94
C.0	Structure of UML Profile for TDL.....	94
C.1	Foundation.....	95
C.1.0	Overview.....	95
C.1.1	Element.....	95
C.1.2	NamedElement.....	95
C.1.3	PackageableElement.....	96
C.1.4	Package.....	96
C.1.5	ElementImport.....	96
C.1.6	Comment.....	97
C.1.7	Annotation.....	97
C.1.8	AnnotationType.....	98
C.1.9	TestObjective.....	98

C.2	Data	98
C.2.1	Data Definition	98
C.2.1.0	Overview	98
C.2.1.1	DataResourceMapping	99
C.2.1.2	MappableDataElement	99
C.2.1.3	DataElementMapping	99
C.2.1.4	ParameterMapping	100
C.2.1.5	DataType	100
C.2.1.6	DataInstance	100
C.2.1.7	SimpleDataType	101
C.2.1.8	SimpleDataInstance	101
C.2.1.9	StructuredDataType	102
C.2.1.10	Member	102
C.2.1.11	StructuredDataInstance	102
C.2.1.12	MemberAssignment	103
C.2.1.13	Parameter	103
C.2.1.14	FormalParameter	103
C.2.1.15	Variable	104
C.2.1.16	Action	104
C.2.1.17	Function	104
C.2.2	Data Use	105
C.2.2.0	Overview	105
C.2.2.1	DataUse	105
C.2.2.2	ParameterBinding	106
C.2.2.3	StaticDataUse	106
C.2.2.4	DataInstanceUse	106
C.2.2.5	SpecialValueUse	107
C.2.2.6	AnyValue	107
C.2.2.7	AnyValueOrOmit	107
C.2.2.8	OmitValue	108
C.2.2.9	DynamicDataUse	108
C.2.2.10	FunctionCall	108
C.2.2.11	FormalParameterUse	109
C.2.2.12	VariableUse	109
C.3	Time	109
C.3.0	Overview	109
C.3.1	Time	111
C.3.2	TimeLabel	111
C.3.3	TimeLabelUse	112
C.3.4	TimeConstraint	112
C.3.5	TimeOperation	112
C.3.6	Wait	113
C.3.7	Quiescence	113
C.3.8	Timer	113
C.3.9	TimerOperation	114
C.3.10	TimerStart	114
C.3.11	TimerStop	114
C.3.12	TimeOut	115
C.4	Test Configuration	115
C.4.0	Overview	115
C.4.1	GateType	116
C.4.2	GateInstance	116
C.4.3	ComponentType	116
C.4.4	ComponentInstance	117
C.4.5	ComponentInstanceRole	117
C.4.6	GateReference	118
C.4.7	Connection	118
C.4.8	TestConfiguration	118
C.5	Test Behaviour	119
C.5.1	Test Description	119

C.5.1.0	Overview	119
C.5.1.1	TestDescription	119
C.5.1.2	BehaviourDescription	120
C.5.2	Combined Behaviour	120
C.5.2.0	Overview	120
C.5.2.1	Behaviour	121
C.5.2.2	Block	121
C.5.2.3	CombinedBehaviour	122
C.5.2.4	SingleCombinedBehaviour	122
C.5.2.5	CompoundBehaviour	122
C.5.2.6	BoundedLoopBehaviour	122
C.5.2.7	UnboundedLoopBehaviour	123
C.5.2.8	MultipleCombinedBehaviour	123
C.5.2.9	AlternativeBehaviour	123
C.5.2.10	ConditionalBehaviour	124
C.5.2.11	ParallelBehaviour	124
C.5.2.12	ExceptionalBehaviour	124
C.5.2.13	DefaultBehaviour	125
C.5.2.14	InterruptBehaviour	125
C.5.2.15	PeriodicBehaviour	125
C.5.3	Atomic Behaviour	126
C.5.3.0	Overview	126
C.5.3.1	AtomicBehaviour	127
C.5.3.2	Break	127
C.5.3.3	Stop	128
C.5.3.4	VerdictAssignment	128
C.5.3.5	Assertion	128
C.5.3.6	Interaction	129
C.5.3.7	Target	129
C.5.3.8	TestDescriptionReference	130
C.5.3.9	ComponentInstanceBinding	130
C.5.3.10	ActionBehaviour	131
C.5.3.11	ActionReference	131
C.5.3.12	InlineAction	132
C.5.3.13	Assignment	132
Annex D (informative):	Bibliography	133
History		134

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (<https://ipr.etsi.org/>).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Methods for Testing and Specification (MTS).

The present document is part 1 of a multi-part deliverable covering the Test Description Language, as identified below:

- Part 1:** "Abstract Syntax and Associated Semantics";
- Part 2: "Graphical Syntax";
- Part 3: "Exchange Format";
- Part 4: "Structured Test Objective Specification (Extension)".

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](#) (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document specifies the abstract syntax of the Test Description Language (TDL) in the form of a meta-model based on the OMG® Meta Object Facility™ (MOF) [1]. It also specifies the semantics of the individual elements of the TDL meta-model. The intended use of the present document is to serve as the basis for the development of TDL concrete syntaxes aimed at TDL users and to enable TDL tools such as documentation generators, specification analyzers and code generators.

The specification of concrete syntaxes for TDL is outside the scope of the present document. However, for illustrative purposes, an example of a possible textual syntax together with its application on some existing ETSI test descriptions are provided.

NOTE: OMG®, UML®, OCL™ and UTP™ are the trademarks of OMG (Object Management Group). This information is given for the convenience of users of the present document and does not constitute an endorsement by ETSI of the products named.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at <http://docbox.etsi.org/Reference>.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

- [1] OMG®: "OMG Meta Object Facility™ (MOF) Core Specification V2.4.1", formal/2013-06-01.

NOTE: Available at <http://www.omg.org/spec/MOF/2.4.1/>.

- [2] OMG®: "OMG Unified Modeling Language™ (OMG UML) Superstructure, Version 2.4.1", formal/2011-08-06.

NOTE: Available at <http://www.omg.org/spec/UML/2.4.1/>.

- [3] OMG®: "Object Constraint Language™ (OCL), Version 2.4", formal/2014-02-03.

NOTE: Available at <http://www.omg.org/spec/OCL/2.4/>.

- [4] Void.

- [5] ETSI ES 203 119-3 (V1.2.1): "Methods for Testing and Specification (MTS); The Test Description Language (TDL); Part 3: Exchange Format".

- [6] ETSI ES 203 119-4 (V1.2.1): "Methods for Testing and Specification (MTS); The Test Description Language (TDL); Part 4: Structured Test Objective Specification (Extension)".

- [7] ISO/IEC 9646-1:1994: "Information technology - Open Systems Interconnection -- Conformance testing methodology and framework -- Part 1: General concepts".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

- [i.1] ETSI ES 201 873-1 (V4.5.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language".
- [i.2] ETSI TS 136 523-1 (V10.2.0): "LTE; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Packet Core (EPC); User Equipment (UE) conformance specification; Part 1: Protocol conformance specification (3GPP TS 36.523-1 version 10.2.0 Release 10)".
- [i.3] ETSI TS 186 011-2: "Core Network and Interoperability Testing (INT); IMS NNI Interoperability Test Specifications (3GPP Release 10); Part 2: Test descriptions for IMS NNI Interoperability".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

abstract syntax: graph structure representing a TDL specification in an independent form of any particular encoding

action: any procedure carried out by a component of a test configuration or an actor during test execution

actor: abstraction of entities outside a test configuration that interact directly with the components of that test configuration

component: active element of a test configuration that is either in the role tester or system under test

concrete syntax: particular representation of a TDL specification, encoded in a textual, graphical, tabular or any other format suitable for the users of this language

interaction: any form of communication between components that is accompanied with an exchange of data

meta-model: modelling elements representing the abstract syntax of a language

system under test (SUT): role of a component within a test configuration whose behaviour is validated when executing a test description

TDL model: instance of the TDL meta-model

TDL specification: representation of a TDL model given in a concrete syntax

test configuration: specification of a set of components that contains at least one tester component and one system under test component plus their interconnections via gates and connections

test description: specification of test behaviour that runs on a given test configuration

test verdict: result from executing a test description

tester: role of a component within a test configuration that controls the execution of a test description against the components in the role system under test

tester-input event: event that occurs at a component in the role tester and determines the subsequent behaviour of this tester component

<undefined>: semantical concept denoting an undefined data value

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ADT	Abstract Data Type
EBNF	Extended Backus-Naur Form
IEC	International Electrotechnical Commission
IMS	IP Multimedia Subsystem
ISO	International Organization for Standardization
MBT	Model-Based Testing
MOF	Meta-Object Facility™
OCL	Object Constraint Language™
OMG	Object Management Group®
SUT	System Under Test
TDD	Test Driven Development
TDL	Test Description Language
TTCN-3	Testing and Test Control Notation version 3
UML	Unified Modelling Language®
URI	Unified Resource Identifier
XML	eXtensible Markup Language

4 Basic Principles

4.1 What is TDL?

TDL is a language that supports the design and documentation of formal test descriptions that may be the basis for the implementation of executable tests in a given test framework, such as TTCN-3 [i.1]. Application areas of TDL that will benefit from this homogeneous approach to the test design phase include:

- Manual design of test descriptions from a test purpose specification, user stories in test driven development or other sources.
- Representation of test descriptions derived from other sources such as MBT test generation tools, system simulators, or test execution traces from test runs.

TDL supports the design of black-box tests for distributed, concurrent real-time systems. It is applicable to a wide range of tests including conformance tests, interoperability tests, tests of real-time properties and security tests based on attack traces.

TDL clearly separates the specification of tests from their implementation by providing an abstraction level that lets users of TDL focus on the task of describing tests that cover the given test objectives rather than getting involved in implementing these tests to ensure their fault detection capabilities onto an execution framework.

TDL is designed to support different abstraction levels of test specification. On one hand, the concrete syntax of the TDL meta-model may hide meta-model elements that are not needed for a declarative (more abstract) style of specifying test descriptions. For example, a declarative test description could work with the time operations *wait* and *quiescence* instead of explicit timers and operations on timers (see clause 9).

On the other hand, an imperative (less abstract or refined) style of a test description supported by a dedicated concrete syntax could provide additional means necessary to derive executable test descriptions from declarative test descriptions. For example, an imperative test description could include timers and timer operations necessary to implement the reception of SUT output at a tester component and further details. It is expected that most details of a refined, imperative test description can be generated automatically from a declarative test description. Supporting different levels of abstraction by a single TDL meta-model offers the possibility of working within a single language and using the same tools, simplifying the test development process that way.

4.2 Design Considerations

TDL makes a clear distinction between concrete syntax that is adjustable to different application domains and a common abstract syntax, which a concrete syntax is mapped to (an example concrete syntax is provided in annex B). The definition of the abstract syntax for a TDL specification plays the key role in offering interchangeability and unambiguous semantics of test descriptions. It is defined in the present document in terms of a MOF meta-model.

A TDL specification consists of the following major parts that are also reflected in the meta-model:

- A test configuration consisting of at least one tester and at least one SUT component and connections among them reflecting the test environment.
- A set of test descriptions, each of them describing one test scenario based on interactions between the components of a given test configuration and actions of components or actors. The control flow of a test description is expressed in terms of sequential, alternative, parallel, iterative, etc. behaviour.
- A set of data definitions that are used in interactions and as parameters of test description invocations.
- Behavioural elements used in test descriptions that operate on time.

Using these major ingredients, a TDL specification is abstract in the following sense:

- Interactions between tester and SUT components of a test configuration are considered to be atomic and not detailed further. For example, an interaction can represent a message exchange, a remote function/procedure call, or a shared variable access.
- All behavioural elements within a test description are totally ordered, unless it is specified otherwise. That is, there is an implicit synchronization mechanism assumed to exist between the components of a test configuration.
- The behaviour of a test description represents the expected, foreseen behaviour of a test scenario assuming an implicit test verdict mechanism, if it is not specified otherwise. If the specified behaviour of a test description is executed, the 'pass' test verdict is assumed. Any deviation from this expected behaviour is considered to be a failure of the SUT, therefore the 'fail' verdict is assumed.
- An explicit verdict assignment may be used if in a certain case there is a need to override the implicit verdict setting mechanism (e.g. to assign 'inconclusive' or any user-defined verdict values).
- The data exchanged via interactions and used in parameters of test descriptions are represented as values of an abstract data type without further details of their underlying semantics, which is implementation-specific.
- There is no assumption about verdict arbitration, which is implementation-specific. If a deviation from the specified expected behaviour is detected, the subsequent behaviour becomes undefined. In this case an implementation might stop executing the TDL specification.

A TDL specification represents a closed system of tester and SUT components. That is, each interaction of a test description refers to one source component and at least one target component that are part of the underlying test configuration a test description runs on. The actions of the actors (entities of the environment of the given test configuration) may be indicated in an informal way.

Time in TDL is considered to be global and progresses in discrete quantities of arbitrary granularity. Progress in time is expressed as a monotonically increasing function. Time starts with the execution of the first ('base') test description being invoked.

The elements in a TDL specification may be extended with tool, application, or framework specific information by means of annotations.

4.3 Principal Design Approach

The language TDL is designed following the meta-modelling approach which separates the language design into abstract syntax and concrete syntax on the one hand, and static semantics and dynamic semantics on the other hand. The abstract syntax of a language describes the structure of an expression defined in the language by means of abstract concepts and relationships among them, where a concrete syntax describes concrete representation of an expression defined in this language by means of textual, graphical, or tabular constructs which are mapped to concepts from the abstract syntax. The semantics describes the meaning of the individual abstract syntax concepts.

The realization of multiple representations by means of different syntactical notations for a single language requires a clear distinction between abstract syntax and concrete syntax. In a model-based approach to language design, the abstract syntax is defined by means of a meta-model. The meta-model of TDL defines the underlying structure of the abstract concepts represented by means of textual, graphical, or tabular constructs, without any restrictions on how these are expressed by means of e.g. keywords, graphical shapes, or tabular headings. The concrete syntax provides means for the representation of the abstract concepts in the form of textual, graphical, or tabular constructs and defines mappings between the concrete representations and the abstract concepts. This approach allow any concrete representation conforming to a given meta-model to be transformed into another representation conforming to that meta-model, such as graphical to textual, textual to tabular, tabular to graphical, etc. The transformations on the concrete syntax level have no impact on the semantics of the underlying abstract syntax concepts.

The semantics of a language is divided into static semantics and dynamic semantics. The static semantics defines further restrictions on the structure of abstract syntax concepts that cannot be expressed in syntax rules. The dynamic semantics defines the meaning of a syntactical concept when it is put into an execution environment.

The four pieces of the TDL design, concrete syntax, abstract syntax, static semantics, dynamic semantics, are mapped to the standards series of TDL as follows (see figure 4.1):

- TDL-MM, part 1: Covers abstract syntax, static semantics and dynamic semantics;
- TDL-GR, part 2: Covers concrete syntax of graphical TDL;
- TDL-XF, part 3: Covers concrete syntax of the XML-based TDL exchange format;
- TDL-TO, part 4: Covers all parts of concrete/abstract syntax and static/dynamic semantics of the TDL Test Objective extension.

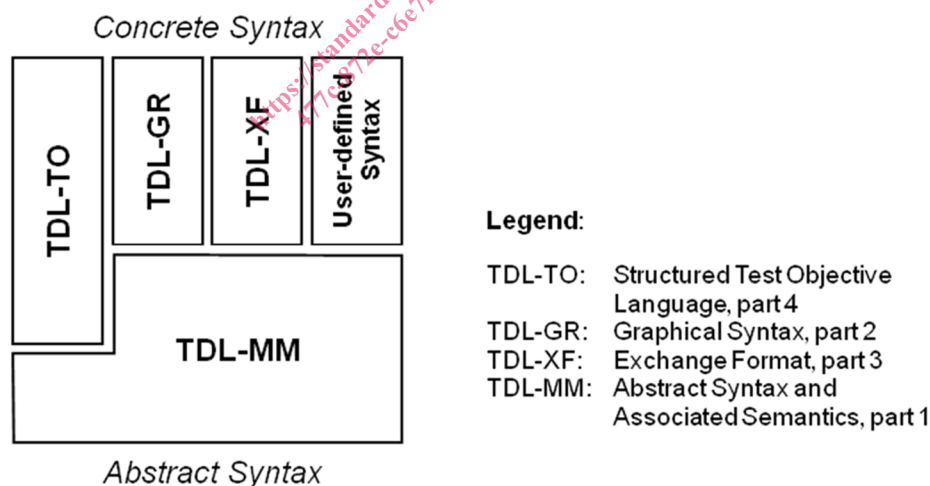


Figure 4.1: The TDL standards and their relation

This decomposition of the TDL language design into the different standard parts allows for the development of integrated and stand-alone tools: editors for TDL specifications in graphical, textual, and user-defined concrete syntaxes, analyzers of TDL specifications that check the consistency of TDL specifications, test documentation generators, test code generators to derive executable tests and others. In all cases the TDL exchange format [5] serves as the bridge between all TDL tools and to ensure tool interoperability (see figure 4.2).