THE ETSI TEST DESCRIPTION LANGUAGE
BRIEF INTRODUCTION

Philip Makedonski (University of Göttingen)
Martti Käärik (Elviour OU)
Overview

What is TDL?

- Test Description Language
  - Design, documentation, and representation of formalised test descriptions
  - Scenario-based approach
- Standardised at ETSI by TC MTS
  - STF 454 (2013)
  - STF 476 (2014)
  - STF 492 (2015-2016)
  - STF 522 (2017)

First steps

- TDL main ingredients
  - Test data
  - Test configuration
  - Test behaviour
  - Test objectives

The TDL Open Source Project
What is TDL?

- Test Description Language
- Design, documentation, and representation of formalised test descriptions
- Scenario-based approach
- Standardised at ETSI by TC MTS
  - STF 454 (2013)
  - STF 476 (2014)
  - STF 492 (2015-2016)
  - STF 522 (2017)
What is TDL?

• Design, documentation, representation?
• ease development and review
• improve productivity and quality
• both industry and standardisation
• reduce implementation details
What is TDL?

- Scenario-based?
  - describe interactions with a system
  - attach test objectives to scenarios
  - derive and automate tests
- Reactive, distributed, real-time
  - common black-box testing concepts
  - domain adaptation
- agile development
What is TDL?

• Standardised?
  • canonical reference
  • stable documentation
  • clear semantics
  • interoperability and independence
  • updated with user needs
  • maintenance commitment
What is TDL?

• Contributions from:
  • Siemens AG, Ericsson Hungary
  • Fraunhofer FOKUS, ETSI CTI
  • CEA, University of Göttingen
  • OU Elvior, Cinderella ApS

• Guidance:
  • Steering Group, TC MTS
What is TDL?

Part 1: MM Meta-Model and Semantics

Part 2: GR Graphical Syntax

Part 3: XF Exchange Format

Part 4: TO Structured Test Objectives

Part 5: UML Profile for TDL

Part 6: Mapping to TTCN-3

Part 7: Extended Test Configurations
What is TDL?
What is TDL?

TDL P1: MM

TDL P2: GR

TDL P3: XF

TDL P4: TO

User-defined syntax

GateType

type [1]

gateInstance [1..*] ordered, unique

ComponentType

timer [*] unique

Variable

variable [*] unique

Foundation

packageableElement

dataType [1..*] unique

Time

Timer

8.2 Abstract Syntax and Classifier Description

8.2.1 GateType

Semantics
A 'GateType' represents a type of communication points, called 'GateInstance's. A 'GateType' specifies the 'DataType's that can be exchanged via 'GateInstance's of this type in both directions.

Generalization
- PackageableElement

Properties
- dataType: DataType [1..*] {unique}
  The 'DataType's that can be exchanged via 'GateInstance's shall adhere to the 'DataType's that are allowed to be exchanged.

Constraints
There are no constraints specified.
6.4.2 GateType

Concrete Graphical Notation

```
context GateType
GateTypeNameLabel := self.name
DataTypeListLabel := self.dataType.name->separator(';', )
```

Comments
No comments.
What is TDL?

TDL P1: MM

TDL P2: GR

User-defined syntax

TDL P3: XF

<xsd:complexType name="ComponentInstance">
    <xsd:complexContent>
        <xsd:extension base="tdl:Element">
            <xsd:choice maxOccurs="unbounded" minOccurs="0">
                <xsd:element name="gateInstance" type="tdl:GateInstance"/>
                <xsd:element name="variable" type="tdl:Variable"/>
            </xsd:choice>
            <xsd:attribute name="componentType" type="xsd:anyURI"/>
            <xsd:attribute name="role" type="tdl:ComponentInstanceRole"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

TDL P4: TO

ComponentInstance

SUT

Tester

role : ComponentInstanceRole

gateInstance [1..*] {ordered, unique}

dataType [1..*] {unique}

PackageableElement

Data::DataDefinition

NamedElement

Data

GateType

GateInstance
What is TDL?

TDL P1: MM

TDL P2: GR

user-defined syntax

TDL P3: XF

<packagedElement xsi:type="tdl:ComponentType">
  <gateInstance xmi:id="_qKt24HasEeWrfP0MdfQNpg" name="g" type="_qKt23nasEeWrfP0MdfQNpg"/>
</packagedElement>

TDL P4: TO
What is TDL?

Base Standard Specification
Identification of Requirements
Creation of ICS/IFS
Definition of TSS
Specification of Test Purposes
Specification of Test Descriptions
Specification of Test Cases
Validation

TDL P4: TO

TP Id | TESTOBJECTIVENAMELABEL
---|---
Test Objective | DESCRIPTIONLABEL
Reference | URIOFOBJECTIVELABEL
Config Id | <CONFIGLABEL>
PICS Selection | <PICSELSELECTIONLABEL>

Initial Conditions

INITIALCONDITIONLABEL

Expected Behaviour

EXPECTEDBEHAVIOURLABEL

Final Conditions

FINALCONDITIONLABEL

© ETSI 2018. All rights reserved
First steps

• TDL main ingredients
  • Test data
  • Test configuration
  • Test behaviour
  • Test objectives
  • Time
First steps

- TDL main ingredients
- Test data
- Test configuration
- Test behaviour
- Test objectives
- Time
First steps

- Test data
  - data definition and data use
  - abstract types and instances
  - composed by using parameters
  - functions and actions
  - mappable to concrete data
  - variables and special values
First steps: Data

Type Login;
Login correct;
Login incorrect;

Use "data.ttcn3" as DATA ;
Map correct to "johnny_correct" in DATA as correct_ttcn3;
Map incorrect to "johnny_incorrect" in DATA as incorrect_ttcn3;

template Login johnny_correct := {
  user := "johnny",
  password := "apple",
  hint := "seed",
  id := 1000
}
template Login johnny_incorrect := {
  user := "johnny",
  password := "orange",
  hint := "second favourite fruit",
  id := 2000
}
First steps: Data

Type Login;
Login correct;
Login incorrect;

Use "data.ttcn3" as DATA ;
Map correct to "johnny_correct" in DATA as correct_ttcn3;
Map incorrect to "johnny_incorrect" in DATA as incorrect_ttcn3;
First steps: Data

Type Login

Login correct; Login incorrect

Use "data.ttcn3" as DATA;

Map correct to "johnny_correct" in DATA as correct_ttcn3;

Map incorrect to "johnny_incorrect" in DATA as incorrect_ttcn3;
First steps

• Test configuration
  • typed components and gates
  • timers and variables
  • connections among gates
  • component roles
First steps: Configuration

Gate Type gt accepts Login, Response;

Component Type ct having {
  gate g of type gt;
}

Test Configuration tc {
  create Tester tester of type ct;
  create SUT sut of type ct;
  connect tester.g to sut.g;
}
First steps

- Test behaviour
  - defines expected behaviour
  - failure upon deviations by default
  - actions and interactions
  - alternative, parallel, iterative, conditional
  - defaulting, interrupting, breaking
First steps: Behaviour

Test Description td (p of type Login)
uses configuration tc {
    tester.g sends incorrect to sut.g;
    alternatively {
        sut.g sends failure to tester.g with {
            test objectives : tp;
        };
        set verdict to pass;
    } or {
        sut.g sends success to tester.g;
        set verdict to fail;
    }
}

or simply (relying on the default semantics):

Test Description td_default (p of type Login)
uses configuration tc {
    tester.g sends incorrect to sut.g;
    sut.g sends failure to tester.g with {
        test objectives : tp;
    };
}
First steps

• Test objectives
  • may be attached to
    • behaviour (atomic or compound)
    • whole test description
  • contain description and reference
First steps: Objectives

Test Objective tp {
    description : "ensure that when incorrect login is provided a failure response is sent";
}

Test Description td (p of type Login)
uses configuration tc {
    tester.g sends incorrect to sut.g;
    alternatively {
        sut.g sends failure to tester.g with {
            test objectives : tp;
        }
        set verdict to pass;
    } or {
        sut.g sends success to tester.g;
        set verdict to fail;
    }
}
First steps

• Structured test objectives
• based on TPLan
• refine test objectives
• formalise specification
• integrate and unify test description and test purpose specification
First steps: Structured Objectives

Test Purpose {

    TP Id "TP/CAM/INA/DOP/BV/02"
    Test objective "Checks that CAM message includes
                   DoorOpen information 30s after closed"
    Reference "TS 102 637-2 [1], clauses 7.1 and 7.2"
    PICS Selection PICS_PUBTRANSVEH
    Initial conditions
    with {
        the IUT entity having reached an initial_state
        and
        the IUT entity having sent a valid CAM message
        containing DoorOpen TaggedValue;
    }
    Expected behaviour
    ensure that {
        when {
            the door entity is closed
        }
        then {
            the IUT entity sends a new CAM message
            containing DoorOpen TaggedValue;
        }
    }
}
What is new in TDL?

- **Part 1: New features**
  - collections, procedures
  - local ordering option
  - UML Profile moved to Part 5
- **Part 6: Mapping to TTCN-3**
  - compatibility and consistency
  - reuse tools and assets
- **Part 7: Extended Configurations**
  - instantiate existing configurations
  - reuse and extend
What is new in TDL?

Test Configuration
compositeTC

Test Configuration
source:defaultTC

TESTER
SS:defaultCT
g
TESTER
Bridge:defaultCT
g
SUT
UE:defaultCT
g

Test Configuration
target:defaultTC

TESTER
SS:defaultCT
g
SUT
UE:defaultCT
g
Mapping TDL to TTCN-3

• Establish a connection between TDL and TTCN-3
• generation of executable tests from test descriptions
• standardised, ensuring compatibility and consistency
• re-use existing tools and frameworks for test execution
• re-use existing TTCN-3 assets (data, behaviour)
Methods for Testing and Specification (MTS); The Test Description Language (TDL); Part 6: Mapping to TTCN-3

ETSI STANDARDS
The TDL Open Source Project

- New technology, growing rapidly
- TDL open source project for essential tool support
  - lower barrier to entry, accelerate adoption
  - commercial tool support not yet available
- Custom tools can be put together in a matter of hours
  - basic yet capable
  - make early adoption easier
- Advanced solutions still require additional effort
  - not immediately necessary to get started with using TDL
The TDL Open Source Project

• Meta-model implementation and validation (Part 1)
• Textual editors (Annex B of Parts 1 and 4)
• Graphical editor (Part 2)
• Translation between representations (Part 3)
• TTCN-3 generation (Part 6, currently ongoing)
The TDL Open Source Project
The TDL Open Source Project
Summary

What is TDL?

- Test Description Language
- Design, documentation, and representation of formalised test descriptions
- Scenario-based approach
- Standardised at ETSI by TC MTS
  - STF 454 (2013)
  - STF 476 (2014)
  - STF 492 (2015-2016)
  - STF 522 (2017)

First steps

- TDL main ingredients
- Test data
- Test configuration
- Test behaviour
- Test objectives

The TDL Open Source Project

tdl.etsi.org