Comparing TTCN-3 and TTCN-2

TTCN-3 User Conference

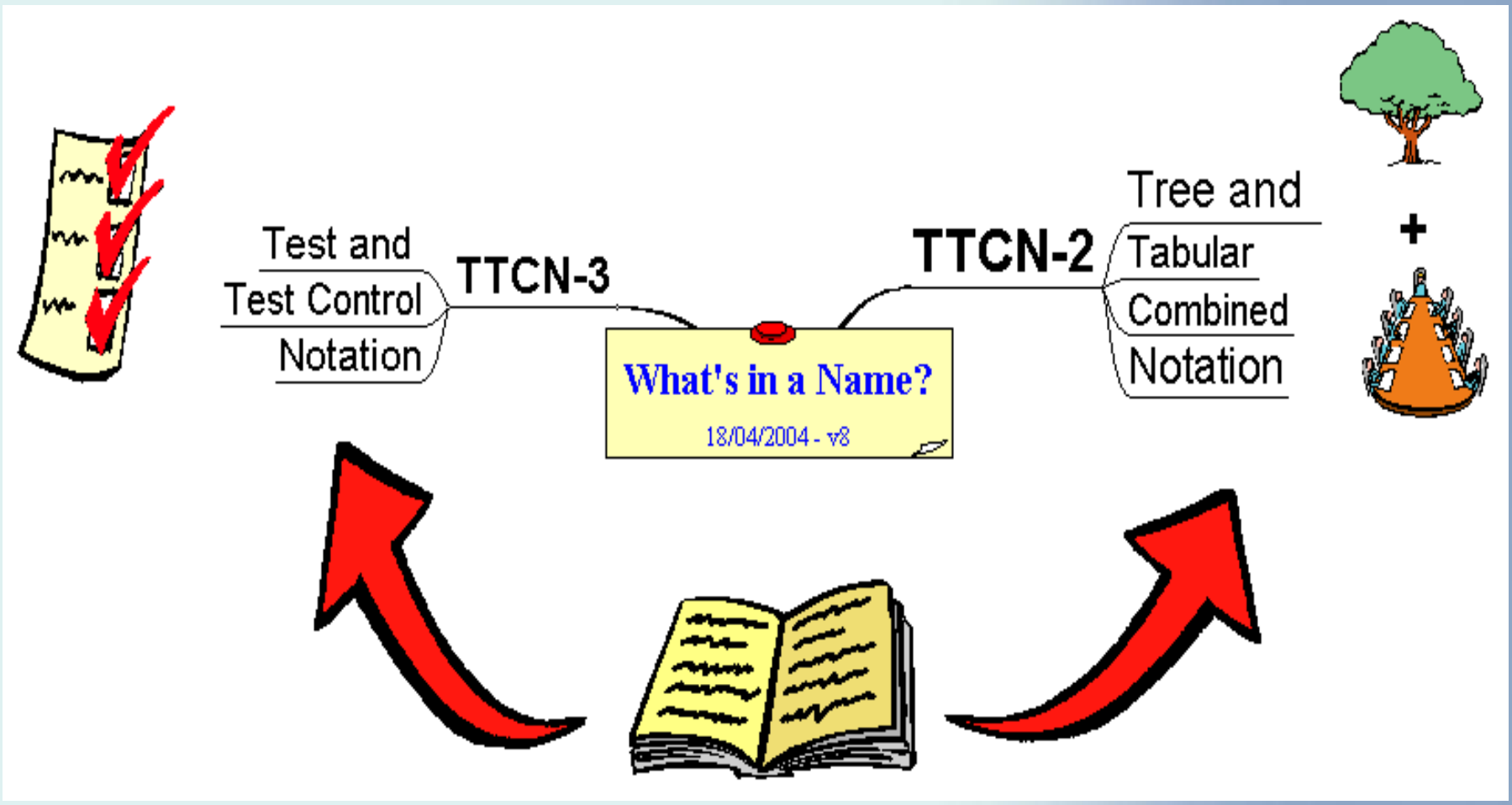
May 3rd-May 5th,2004

Sophia Antipolis

Introduction

- Standards
- Presentation Formats
- Structure
- Types
- Comparing TTCN-3 and TTCN-2 syntax!
- Migrating to TTCN-3?
- Conclusions

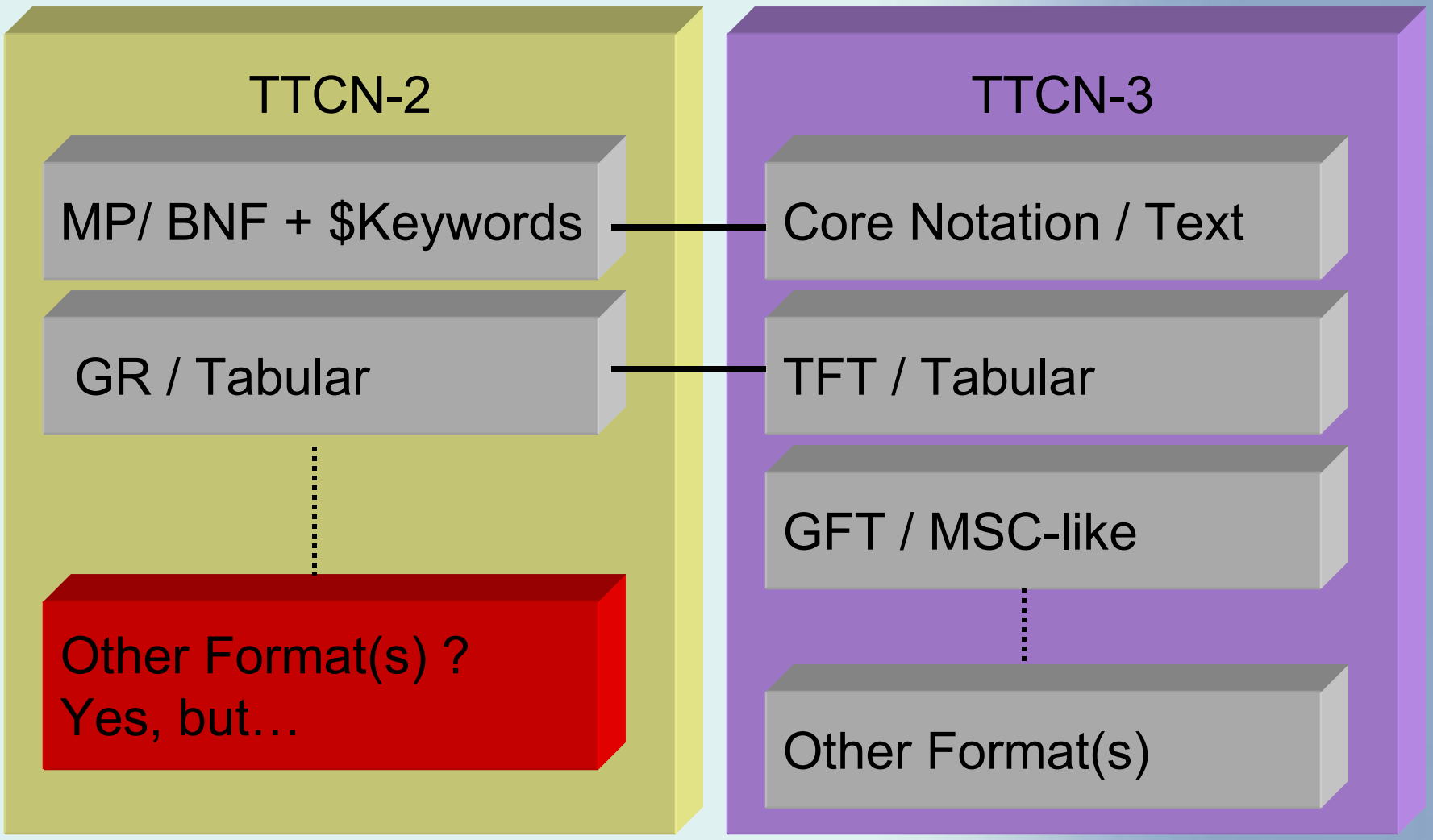
What's in a Name?



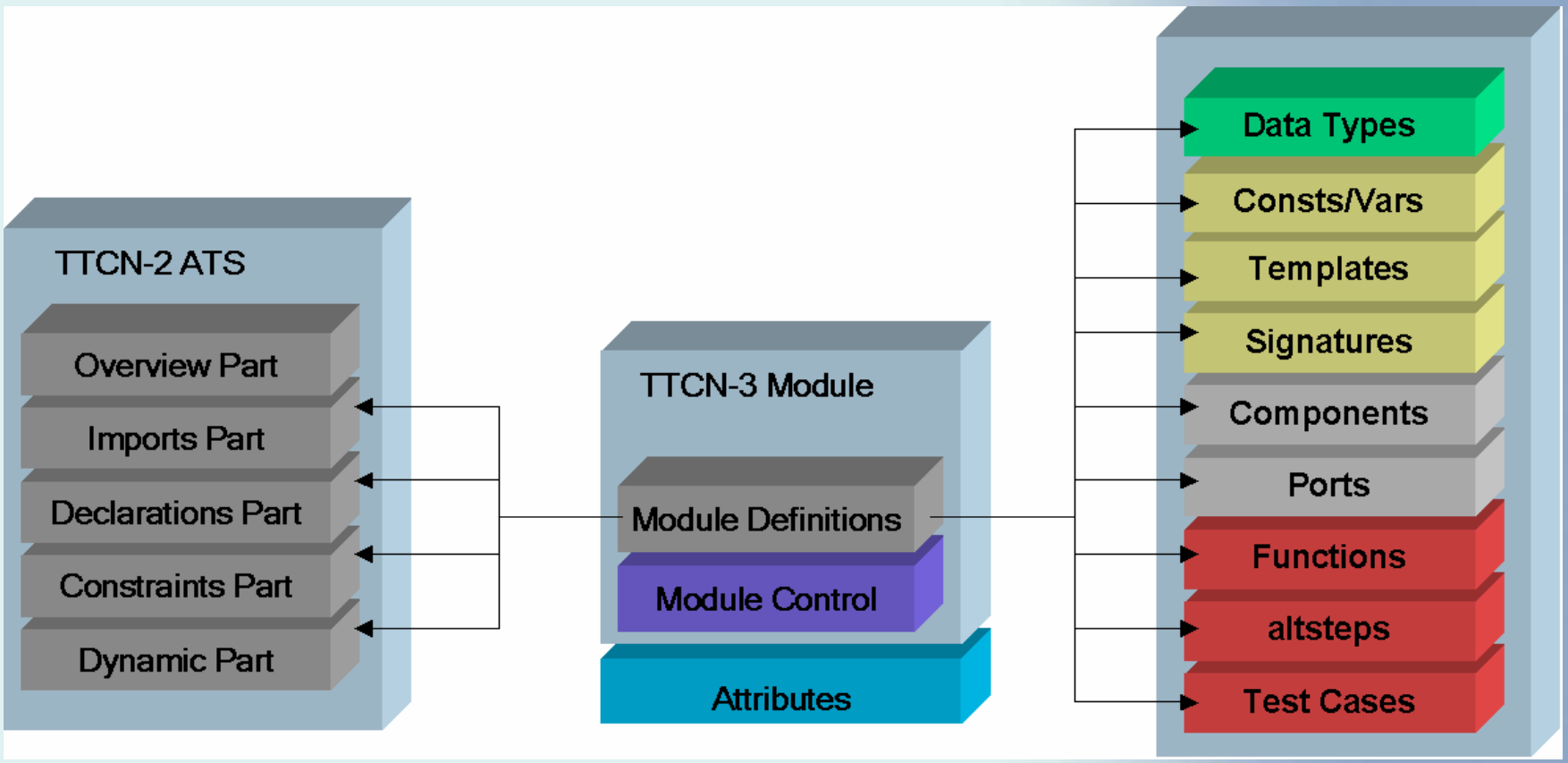
Standards

TTCN-2		
ETSI	**ISO/ITU-T**	
TR101 666	IS9646-3/ X.292	The Tree and Tabular Combined Notation (TTCN) Ed 2++
TTCN-3		
ETSI	**ITU-T**	
ES 201 873-1	Z.140	Part 1: TTCN-3 Core Language.
ES 201 873-2	Z.141	Part 2: TTCN-3 Tabular Presentation Format (TFT)
ES 201 873-3	Z.142	Part 3: TTCN-3 Graphical Presentation Format (GFT).
TR 201 873-4	Z.143	Part 4: TTCN-3 Operational Semantics.
ES 201 873-5	Z.144	Part 5: TTCN-3 Runtime Interface (TRI)
ES 201 873-6	Z.145	Part 6: TTCN-3 Control Interface (TCI)
TR101-874	-	TTCN-3: TTCN-2 to TTCN-3 Mapping (incomplete details)

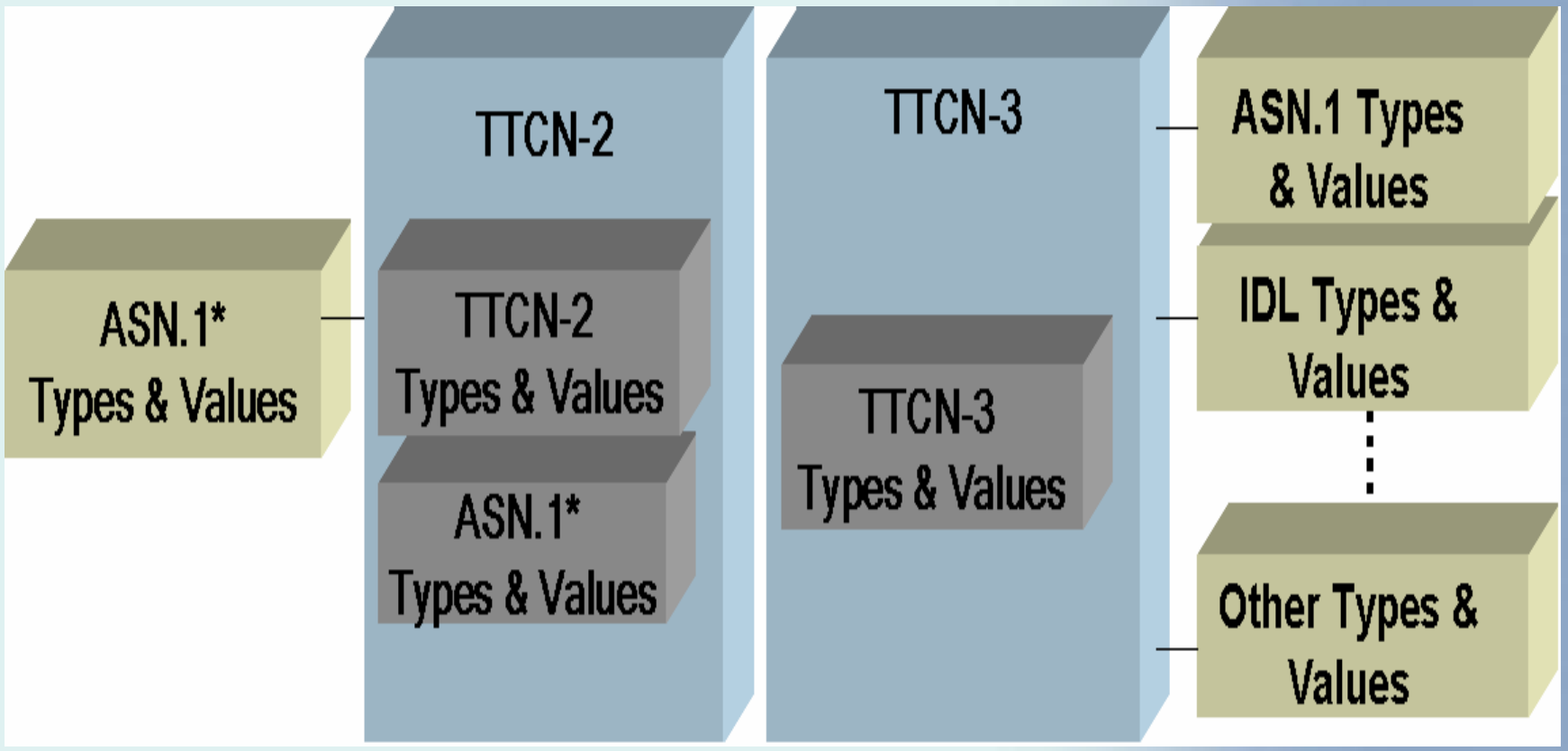
Presentation Formats



Structure


TTCN-2 ATS
Overview Part
Imports Part
Declarations Part
Constraints Part
Dynamic Part
TTCN-3 Module
Module Definitions
Module Control
Attributes
Data Types
Consts/Vars
Templates
Signatures
Components
Ports
Functions
altsteps
Test Cases

Types


TTCN-2
ASN.1*
Types & Values
TTCN-2
Types & Values
ASN.1*
Types & Values
TTCN-3
TTCN-3
Types & Values
ASN.1 Types
& Values
IDL Types &
Values
Other Types &
Values

Types

- TTCN-2 & TTCN-3 are weakly typed languages
- TTCN-3 provides extensive type compatibility rules
- TTCN-3 adds several new types
  - For convenience
  - For better harmonization with ASN.1

Basic Types

Class of type	TTCN-3	TTCN-2
Simple basic types	**integer**	INTEGER
	float	-
	boolean	BOOLEAN
	objid	OBJECTID
	verdicttype	R_type
Basic string types	**bitstring**	BITSTRING
	hexstring	HEXSTRING
	octetstring	OCTETSTRING
	charstring	-
	universal charstring	-
Structured types	**record**	Structure type (PDU,ASP,CM)
	record of	-
	set	-
	set of	-
	enumerated	-
	union	PDU (subset)
Special data types	**anytype**	-
Special configuration types	**address**	-
	port	PCO, CP
	component	(generic)
Special default types	**default**	-

New TTCN-3 Types

- **address**—provides access to SUT internals.
- **port**— replaces TTCN-2 PCO and CP types
- **component**— adds the concept of component types
- **record**— replaces TTCN-2 structured type, ASN.1 type, and tabular and ASN.1 ASP, PDU and CM types (8 types in all) (no implicit fields)
- "Arrays" — Although strictly not a type, single and multi-dimensional support for arrays was added

New Types for use with ASN.1

- Adds:
  - char, charstring, universal charstring,
  - record [of] (SEQUENCE [OF])
  - set [of] (SET [OF])
  - union (CHOICE)
  - float (REAL)
  - enumerated (ENUMERATED)

Parameterization

TTCN-2		TTCN-3		
Element	**Pars**	**Element**	**Equiv**	**Permitted**
Test Suite Operations				
Predefined TSO	**CBV**	**function →predefined**	**in**	**in**
TSOP	**CBV**	**function → user defined**	**in**	**in, out, inout**
TSO	**CBV**	**external function**	**in**	**in, out, inout**
Constraint	**CBV**	**template**	**in**	**in**
Encoding variation	**CBV**	**with { variant "variant rule" }**	**in**	**not possible**
Invalid encoding	**CBV**	**function**	**in**	**in, out, inout**
Global Test Steps	**CBR[1]**	**altstep**	**in[a],inout**	**in, out, inout**
Local Test Steps[2]	**CBR[3]**	**altstep**	**in[b], inout**	**in, out, inout**
Default Step	**CBR**	**default altstep (used in activate only)**	**in[4]**	**in, out, inout[5]**

1 The call by value (CBV) mechanism is used for test steps (global or local) with formal parameters when they are referenced in a CREATE construct;

2 TTCN-3 does not support local altsteps. This would be a useful addition;

3 TTCN-3 modules and test cases can be parameterised;

4 TTCN-2 uses CBR and TTCN-3 uses CBV;

5 Defaults must be replaced with direct altstep references to provide equivalency.

Scope

- TTCN-2: Two levels of scope
  - Global for TSVs*, TSCs, TSPs, TCVs*, and
  - Local for variables in TSOPs & IFEOD
- TTCN-3: Seven levels of scope
  - Module
  - Control Part
  - Components
  - Functions
  - Altsteps
  - Test cases and
  - Blocks of statements

TSV: Test Suite Variable
TCV: Test Case Variable
TSC: Test Suite Constant
TSP: Test Suite Parameter
TSOP: Test Suite Operation Procedure
IFEOD: Invalid Field Encoding Operation Description
* Exceptions not visible in:
- constraints,
- TSOPs,
- IFEODs

Change = Big Returns?

- Modernizing: New look & feel
  - Lower case reserved words→improved readability
  - Object Oriented-like syntax
  - Clear separation of TTCN-3 & ASN.1
  - Removing cryptic & OSI centric notation
  - Eliminating duplicate information
  - Left to right order of evaluation
  - Simpler verdict assignment rules

Timers

Timer Name	Duration	Unit[1]	Comment
MyTimer	1	*ms*	Duration: integer (optional) Unit: *ps*, *ns*, *us*, *ms*, s, *min* (mandatory).

[1] Time units include: *ps* (picosecond), *ns* (nanosecond), *us* (microsecond), *ms* (millisecond), *s* (second) and *min* (minute).

```
timer myTimer := 0.001; // TTCN-3 timer 1ms
```

TTCN-3 timers always have a base unit of second

Send & Receive

TTCN-2 Send

Nr	L	Behaviour Description	Cref	V	C
1		p ! pduX [a = 5] (pduX.x := 4) START T1	c_X	(P)	

TTCN-3 Send

```
[a == 5] p.send(modifies {c_X:= x 4}){ T1.start; setverdict(pass); }
```

TTCN-2 Receive

Nr	L	Behaviour Description	Cref	V	C
1		p ? pduX [a =5] (x:= 4) CANCEL T1	c_X	(P)	

TTCN-3 Receive

```
[ a == 5] p.receive (c_X){ x := 4; T1.stop; setverdict(pass); }
```

Verdicts

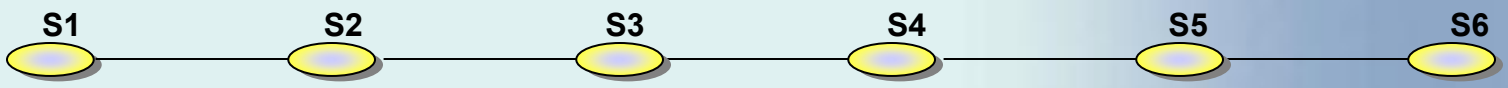
Current value of Verdict	New verdict assignment value			
	pass	inconc	fail	none
none	pass	inconc	fail	none
pass	pass	inconc	fail	pass
inconc	inconc	inconc	fail	inconc
fail	fail	fail	fail	fail

TTCN-3 adopts the preliminary verdicts as used in TTCN-2.

TTCN-3 adds the ‘error’ verdict which can only be set by the test System (caused by a test case execution). Cannot be set by use of setverdict();

NOTE: TTCN-3 does not provide the global R variable as in TTCN-2. Care must be taken if this is used in a test suite.

Sequential Behaviour



TTCN-2

```
S1
  S2
    S3
      S4
        S5
          S6
```

TTCN-2 uses indentation

TTCN-3

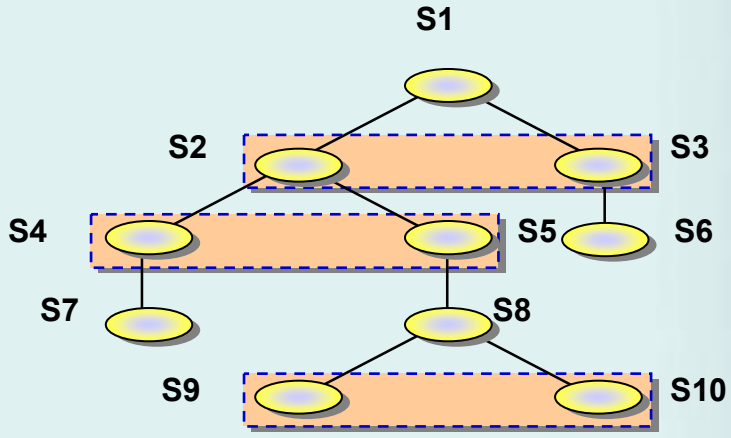
```
S1;
   S2;
      S3;
         S4;
            S5;
              S6;

Or preferably:

S1; S2;  S3;  S4;  S5;  S6
```

TTCN-3 is a free format notation

Alternative Behaviour



TTCN-2

```
S1
  S2
    S4
      S7
    S5
      S8
        S9
        S10
   S3
    S6
```

S2&S3, S4&S5, and S9&S10 are alternatives

S1,S2,S4,S7.
S1,S2,S5,S8,S9.
S1,S2,S5,S8,S10. and
S1,S3,S6. are the possible event sequences.

TTCN-3

```
S1; alt {
      [] S2;alt {
              [] S4; S7;

              [] S5; S8; alt {
                           [] S9;
                           [] S10;
                         }
            }
      [] S3; S6;
    }
```

Why Migrate to TTCN-3?

- TTCN-3 is superior to TTCN-2
- Number of tool vendors to choose from increasing
- Good tools exist for:
  - Development
  - Conversion from TTCN-2 to TTCN-3

Why Migrate to TTCN-3?

- TTCN-3's new look&feel and Java/C++ like syntax is expected to lower developers & testers resistance to its use.
- Which:
  - Should help break down the usually disjoint product development/test process
  - Should encourage re-use, sharing of tests, and improve the team's overall product knowledge
  - Should improve team flexibility (developers/testers may be interchanged)
  - Should lead to shorter test suite development time

Conclusions

- TTCN-3 is a superior solution to TTCN-2
- TTCN-2 will continue to exist for a few years
- Migration from TTCN-2 to TTCN-3 will be gradual
- To ensure TTCN-3's evolution and continued success tool vendors must:
  - Provide input to, and offer continued support to standards bodies,
  - Provide affordable, innovative, & timely solutions which meet user needs,
  - Offer user friendly, simple to use, & high quality tools,
  - Promote use of TTCN-3 by:
    - Keeping their tools aligned to the evolving standards.
    - Offering professional training and quality support.
- TTCN-3 provides organizations with the ability to design and architect well engineered test systems!