ERICSSON

TITAN 6.5.0 CHANGE LOG

2018.12.04

NEW COMPILER/MAKEFIELGEN OPTIONS

- Added compiler option '-I', which enables the real-time testing features mentioned here. The features are disabled by default, and the new keywords ('now', 'realtime' and 'timestamp') can be used as identifiers again (for backward compatibility).

- Also added makefilegen option '-i', which activates this option for the compiler, and the makefile setting 'enableRealtimeTesting' in the TPD, which does the same thing.

NEW FLAG FOR XSD2TTCN

- xsd2ttcn
- -o: generate all definitions into one module (called XSD_Definitions)

NEW PRODUCTS ADDED

- git://git.eclipse.org/gitroot/titan/titan.Servers.GTP_Tunnel_Daemon
- git://git.eclipse.org/gitroot/titan/titan.Servers.IP_Daemon_Dynamic
- git://git.eclipse.org/gitroot/titan/titan.Servers.SCTP_Daemon_Dynamic

- git://git.eclipse.org/gitroot/titan/titan.ProtocolEmulations.SCTP

- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.NAS_EPS_15.2.0.1
- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.GTPv2_v15.2.0
- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.PFCP_v15.1.0
- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.V5
- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.STUN
- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.SGsAP_13.2.0
- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.XML_RPC

- git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.GCP_31r1

- git://git.eclipse.org/gitroot/titan/titan.TestPorts.Thrift_TPG

DOCUMENTATION MIGRATED TO ASCIIDOC

CRL 113 200/5	TITAN TTCN-3 EXECUTOR Eclipse Plug-ins	https://github.com/eclipse/titan.EclipsePlug-ins
CRL 113 200/6	TITAN TTCN-3 EXECUTOR	https://github.com/eclipse/titan.core
CNL 113 512	EPTF_Core_Library	git://git.eclipse.org/gitroot/titan/titan.Libraries.CLL.git

DOCUMENTATION MIGRATED TO ASCIIDOC

CNL 113 312	HTTP TEST PORT	https://github.com/eclipse/titan.TestPorts.HTTPmsg
CNL 113 319	SIP TEST PORT	https://github.com/eclipse/titan.TestPorts.SIPmsg
CNL 113 320	TELNET TEST PORT	https://github.com/eclipse/titan.TestPorts.TELNETasp
CNL 113 334	PIPE TEST PORT	https://github.com/eclipse/titan.TestPorts.PIPEasp
CNL 113 337	MTP3 M3UA TEST PORT	git://git.eclipse.org/gitroot/titan/titan.TestPorts.MTP3.git
CNL 113 341	SCCP PROTOCOL EMULATIONS	git://git.eclipse.org/gitroot/titan/titan.ProtocolEmulations.SCCP.git
CNL 113 346	UDP SOCKET	https://github.com/eclipse/titan.TestPorts.UDPasp
CNL 113 347	TCP SOCKET	https://github.com/eclipse/titan.TestPorts.TCPasp
CNL 113 353	SDP PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.SDP.git
CNL 113 365	ISUP Q.762 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.ISUP_Q.762.git
CNL 113 368	COMMON PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.COMMON
CNL 113 384	ABSTRACT SOCKET	https://github.com/eclipse/titan.TestPorts.Common_Components.Abstract_Socket
CNL 113 385	LDAP TEST PORT	https://github.com/eclipse/titan.TestPorts.LDAPmsg
CNL 113 392	RTP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.RTP
CNL 113 418	IP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.IP
CNL 113 420	UDP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.UDP
CNL 113 424	H248 V2 PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.H248_v2
CNL 113 435	DSS1 ETSI PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.DSS1_ETSI.git
CNL 113 439	IUA PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.IUA
CNL 113 443	PCAP TEST PORT	https://github.com/eclipse/titan.TestPorts.PCAPasp
CNL 113 461	DHCP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.DHCP
CNL 113 462	DIAMETER PROTOCOL MODULE GENERATOR	https://github.com/eclipse/titan.ProtocolModules.DIAMETER_ProtocolModule_Generator
CNL 113 465	M2UA PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.M2UA.git
CNL 113 467	MSRP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.MSRP
CNL 113 469	SCTP (KERNEL) TEST PORT	https://github.com/eclipse/titan.TestPorts.SCTPasp
CNL 113 484	SSHCLIENT TEST PORT	https://github.com/eclipse/titan.TestPorts.SSHCLIENTasp
CNL 113 493	SUNRPC TEST PORT	https://github.com/eclipse/titan.TestPorts.SUNRPCasp
CNL 113 513	LDAP RFC4511 TEST PORT	https://github.com/eclipse/titan.TestPorts.LDAPasp_RFC4511
CNL 113 517	SUA PROTOCOL EMULATIONS	git://git.eclipse.org/gitroot/titan/titan.ProtocolEmulations.SUA.git
CNL 113 519	LANL2 TEST PORT	https://github.com/eclipse/titan.TestPorts.LANL2asp
CNL 113 529	ICMP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.ICMP
CNL 113 531	IPL4 TEST PORT	https://github.com/eclipse/titan.TestPorts.IPL4asp
CNL 113 536	M3UA PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.M3UA
CNL 113 537	M3UA PROTOCOL EMULATION	git://git.eclipse.org/gitroot/titan/titan.ProtocolEmulations.M3UA.git
CNL 113 576	SNDCP V7.0.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.SNDCP_v7.0.0.git
CNL 113 577	LLC V7.1.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.LLC_v7.1.0.git
CNL 113 578	NS V7.3.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.NS_v7.3.0.git
CNL 113 580	BSSAPP V7.3.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.BSSAPP_v7.3.0.git

DOCUMENTATION MIGRATED TO ASCIIDOC

CNL 113 588	RTSP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.RTSP
CNL 113 598	SMTP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.SMTP
CNL 113 599	PPP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.PPP
CNL 113 600	RADIUS PROTOCOL MODULE GENERATOR	https://github.com/eclipse/titan.ProtocolModules.RADIUS_ProtocolModule_Generator
CNL 113 603	L2TP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.L2TP
CNL 113 631	ICMPV6 PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.ICMPv6
CNL 113 642	STDINOUT TEST PORT	https://github.com/eclipse/titan.TestPorts.STDINOUTmsg
CNL 113 660	IMAP 4REV1 PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.IMAP_4rev1
CNL 113 675	TCP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.TCP
CNL 113 686	SOCKET API	https://github.com/eclipse/titan.TestPorts.Common_Components.Socket-API
CNL 113 702	UNIX DOMAIN SOCKET TEST PORT	https://github.com/eclipse/titan.TestPorts.UNIX_DOMAIN_SOCKETasp
CNL 113 722	EAP PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.EAP.git
CNL 113 760	SQL TEST PORT	https://github.com/eclipse/titan.TestPorts.SQLasp
CNL 113 761	BSSMAP V11.2.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.BSSMAP_v11.2.0.git
CNL 113 763	DHCPV6 PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.DHCPv6
CNL 113 772	SMPP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.SMPP
CNL 113 774	SNMP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.SNMP
CNL 113 775	XMPP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.XMPP
CNL 113 779	INTERNET CONTENT ADAPTION PROTOCOL (ICAP) PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.ICAP
CNL 113 782	WEBSOCKET PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.WebSocket
CNL 113 790	FRAMERELAY PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.FrameRelay
CNL 113 795	PROTOBUFF PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.ProtoBuff
CNL 113 796	HTTP PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.HTTP2.0
CNL 113 801	IKEV2 PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.IKEv2
CNL 113 806	TLS PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.TLS.git
CNL 113 809	IPSEC PROTOCOL MODULE	https://github.com/eclipse/titan.ProtocolModules.IPsec
CNL 113 829	COAP PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.CoAP.git
CNL 113 831	MQTT PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.MQTT.git
CNL 113 832	MOBILEL3 V13.4.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.MobileL3_v13.4.0.git
CNL 113 833	BSSGP V13.0.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.BSSGP_v13.0.0.git
CNL 113 839	TLS (LIBRARY)	git://git.eclipse.org/gitroot/titan/titan.Libraries.TLS.git
CNL 113 843	GTP V13.5.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.GTP_v13.5.0.git
CNL 113 846	GTPV2 V13.7.0 PROTOCOL MODULE	git://git.eclipse.org/gitroot/titan/titan.ProtocolModules.GTPv2_v13.7.0.git
CNL 113 864	EPTF_Web_Gui	git://git.eclipse.org/gitroot/titan/titan.Libraries.ServiceFramework.git
CNL 113 865	EPTF_Service Framework	git://git.eclipse.org/gitroot/titan/titan.Libraries.Web_GUI.git

BUG 521417 - ASN.1 BITSTRING SUBTYPING FALSE ERROR (1)

- ETSI is changing ITS security definitions to ASN.1, this problem come up during this work.
- In module IEEE1609dot2 line 287, a BITSTRING is subtyped, disallowing using an empty named bit list for its instances. This is a legal notation, which is misinterpreted by Titan.
- Titan issues the error messages to this line:
- ../asn1/LibIts/Security/IEEE1609dot2.asn:287: error: syntax error, unexpected $end, expecting <upperidentifier> or <loweridentifier> or {block} or ...
- ../asn1/LibIts/Security/IEEE1609dot2.asn:287: error: TableConstraint can only be applied to ObjectClassFieldType

- In ASN.1, this kind of subtyping is allowed:
- ConstrainedType ::= Type Constraint | <etc.>
- Constraint ::= "(" ConstraintSpec ExceptionSpec ")"
- ConstraintSpec ::= SubtypeConstraint | <etc.>
- SubtypeConstraint ::= ElementSetSpecs
- ElementSetSpec ::= Unions | ALL Exclusions
- Exclusions ::= EXCEPT Elements
- Elements ::= SubtypeElements | <etc.>
- SubtypeElements ::= SingleValue | <etc.>
- SingleValue ::= Value
- Value ::= BuiltinValue
- BuiltinValue ::= BitStringValue | <etc.>
- BitStringValue ::=
- bstring
- | hstring
- | "{" IdentifierList "}"
- | "{" "}"
- | CONTAINING Value

BUG 521417 - ASN.1 BITSTRING SUBTYPING FALSE ERROR (2)

- Changed the parsing of blocks inside type constraints (the block is now processed as a single value constraint instead of a table constraint if it is part of a set operation constraint, such as '... EXCEPT ...').
- Also changed the conditions for merging type constraints (both for merging multiple constraints of one type, and for merging a type's constraint with its parent type's constraint). A constraint no longer has to be a subset of the type's previous constraints or the parent type's constraints, it is now enough if the intersection of the constraints is not empty (meaning at least one value satisfies the merged type constraint). Error message texts have been adjusted accordingly.
- These changes cause the type specified in the example to be accepted by the compiler, however, the compiler does not take other type constraints (such as size constraints) into account when processing a single value constraint of named bits. This means that, in the given example, 'ALL EXCEPT {}' is currently converted to 'ALL EXCEPT ''B' instead of 'ALL EXCEPT '00000000'B' (because 'SIZE (8)' is currently ignored when processing the named bits).
- Regression tests added to regression_test/ASN1/parse/Test1.asn.
- Other tests (negative tests) have also been adjusted to match the type constraint merging change.
- **The mentioned changes have been reverted due to negative side effects.**

BUG 521417 - ASN.1 BITSTRING SUBTYPING FALSE ERROR (3)

- The parsing changes from the previous solution have been re-added.
- The condition for merging 'ALL EXCEPT ...' constraints to a type's other constraints has been changed to the one in the previous solution (intersection with previous constraints is not empty). The conditions for merging all other constraints within a type, and for merging a type's constraints with the parent type's remain the same (new constraint has to be a subset of the type's previous constraints or the parent type's constraints).
- Re-added regression tests to regression_test/ASN1/parse/Test1.asn. All other tests remain the same.

BUG 535441 - FATAL ERROR WHEN USING EXTENDED COMPONENT AS ACTUAL PARAMETER

- The compiler crashes with a fatal error, if a variable of an extended component type is used as actual parameter, where the formal parameter is of the base component type.

- Example:
- module comp_param {

- type component CT_Base {}

- type component CT_Extended extends CT_Base {}

- function f_comp(inout CT_Base p) {}

- control {
- var CT_Extended c;
- f_comp(c);
- }

- }
  Fixed.
- Problem: the reverse-order type compatibility check, which in other cases is needed to make sure the value can be converted back to the original type, failed, since the extended component cannot be converted to its base type.
- Solution: the check was removed if the parameters are of component types, since these don't require type conversion at run-time anyway (there are no classes generated for component types, they all use the built-in class 'COMPONENT').

- Regression tests added under regression_test/compileonly/componentParameter.

BUG 535558 - XML ENC/DEC: ANY ELEMENT NAMESPACE RESTRICTIONS SHOULD NOT AFFECT SUB-ELEMENTS

- Currently the XML codec displays an error if one of the sub-elements of an 'anyElement' does not meet the namespace restrictions set in the coding instructions.
- This is not the desired behavior. Only the top-level element in the 'anyElement' value has to meet the namespace restrictions.

- Example:
- type record Rec {
- universal charstring elem
- }
- with {
- encode "XML";
- variant (elem) "anyElement except unqualified"
- }

- Encoding
- { elem := "<ns:top xmlns:ns='www.somewhere.com'>₁₂₃</ns:top>" }
- currently causes an error, because element 'sub' does not meet the namespace requirements (it's from the unqualified namespace). This restriction should only apply to the top-level element 'top'.
  Fixed.
- The namespace restrictions now only apply to the top-most element in the 'anyElement' value.

- Regression test case added to regression_test/XML/EXER-whitepaper/AnyElement.ttcnpp.

BUG 535729 - FAULT IN RE-ARRANGEMENT OF TEMPLATE DEFAULT PARAMETERS (RT2 ONLY)

- The following code causes a dynamic test case error in Runtime2:

```
type record R1 {
  integer f1,
  boolean f2
}

type record R2 {
  integer f1,
  R1 f2
}

type record R3 {
  charstring f1,
  union {
    integer a1,
    R2 a2
  } f2
}

template R3 t_pard2(template charstring p1,
                    template integer p2) := {
  f1 := p1,
  f2 := {
    a2 := t_pard(p2)
  }
}

template R2 t_pard(template integer p1,
                   template R1 p2 := t_default) := {
  f1 := p1,
  f2 := p2
};
```

BUG 535729 - FAULT IN RE-ARRANGEMENT OF TEMPLATE DEFAULT PARAMETERS (RT2 ONLY)

```
template R1 t_default := {
  f1 := 10,
  f2 := true
};

control {
  var R2 x := valueof(t_pard(21));
  // Dynamic test case error: Copying an uninitialized/unsupported record/set template.
  action(x);
}
```

- The fault lies in the re-arrangement of the template declarations (both t_pard and t_default are used before they are declared, which is allowed in TTCN-3, but not in C++, so the compiler has to re-order them). This causes the initialization code of t_default to be generated to the wrong place (inside the initialization of t_pard2, instead of the module's initialization function), leaving it uninitialized when it is used from the control part.

- This fault is a side-effect of the change made in Runtime2 to support the use of non-deterministic functions as default parameters. See https://bugs.eclipse.org/bugs/show_bug.cgi?id=529019
- Fixed.
- If the default value is a reference to anything other than a function, external function or parameterized template (these are the only references that can be non-deterministic), then only the reference name is generated into the C++ code, not the default parameter's initialization code.

- Test case added to regression_test/nondeterministicDefaultParam/TestsRT2.ttcn.

BUG 535834 - INCORRECT DECODING IS NOT INDICATED IN A CORNER CASE

- In the below example a record is encoded into octetstring and decoded from octetstring, both via a encode/decode style external function and using decmatch templates.
- However the result is slightly different: while the decoding via external function completes successfully, the decmatch reports a remaining octet and fails to match.

- In this case the decmatch is working correctly.
- As the record is encoded into 17 bits ... it is automatically stored on 3 octetst (needed to store it). But when decoding using the decode style external function it fails to notice that after decoding 17 bits correctly there are still 7 bits left from the last octet.

- In this case the encoding/decoding description is incorrect (should be explicitly padded out to fit 3 octets).
- And so the decode style external function should report the warning too.

- The code:
- "
- type component CT{}
- //See RAW_struct_test
- type bitstring BS1 length(1) with { encode "RAW"; variant "FIELDLENGTH(1)" }

- type record PDU101 {
- integer i,
- BS1 bs,
- octetstring os
- } with { encode "RAW"; variant "PADDING(yes)"}

BUG 535834 - INCORRECT DECODING IS NOT INDICATED IN A CORNER CASE

- external function enc_PDU101(in PDU101 pdu) return octetstring
- with { extension "prototype(convert) encode(RAW)"}
- external function dec_PDU101(in octetstring os) return PDU101
- with { extension "prototype(convert) decode(RAW)" }

- type record Message1 {
- integer id,
- octetstring payload
- }

- template Message1 t_Message1 := {
- id := ?,
- payload := decmatch PDU101:{i:=1, bs:=?,os:=?}
- }

- testcase tc_PDU101_1() runs on CT {
- var PDU101 vl_1 := { i:= 1, bs:='1'B, os:= '01'O} //encoded: '010300'O;
- var octetstring vl_encoded := enc_PDU101(vl_1);
- var PDU101 vl_decoded_pdu:= dec_PDU101(vl_encoded);
- var Message1 vl_msg := { id:= 1, payload := enc_PDU101(vl_1)}
- log("vl_msg: ",vl_msg);
- log("vl_decoded_pdu: ", vl_decoded_pdu);
- //log("t_Message1:", t_Message1);
- if(match(vl_msg,t_Message1)) {
- setverdict(pass)
- } else {
- setverdict(fail, match(vl_msg,t_Message1))
- }
- }“
- The "variant "PADDING(yes)"" attribute shows how to handle this case correctly.

- To reproduce the problem, this should be removed.
- Fixed.
- The condition to display the warning for decoder external functions was changed.

BUG 532595 - RAW ENCODER SHOULD NOT ACCEPT INTEGER NUMBERS OUT OF RANGE

- Created attachment 273189 [details]
- RAW encode test for too big integers

- In case of definitions as follows, the range of the acceptable arguments of the encoder function is -32767<=x<=+32767. Numbers out of this range should not be accepted, and a DTE should be thrown.

- Code snippet:
- type integer PDU43 with {variant "16 bit"} // byteorder last, signbit

- external function enc_PDU43(in PDU43 pdu) return octetstring
- with { extension "prototype(convert) encode(RAW)"}

- external function dec_PDU43(in octetstring os) return PDU43
- with { extension "prototype(convert) decode(RAW)" }

BUG 532595 - RAW ENCODER SHOULD NOT ACCEPT INTEGER NUMBERS OUT OF RANGE

- Java code corrected the faults. One trick is here. Signed bit requires an extra bit.
- For testcase tc_raw_int_16bit_wrong1() this helps:
- cpp code (Integer.cc:1177):

```
  else { // not IntX, use the field length
    length = (p_td.raw->fieldlength + 7) / 8;
    if (min_bits(value) > p_td.raw->fieldlength) {
      TTCN_EncDec_ErrorContext::error(TTCN_EncDec::ET_LEN_ERR,
        "There are insufficient bits to encode '%s' : ", p_td.name);
      value = 0; // substitute with zero
    }
  }
```

- java code (TitanInteger.java:939):

```
                                        } else { // not IntX, use the field length
                                                length = (p_td.raw.fieldlength + 7) / 8;
                                                int min_bits = RAW.min_bits(value);
                                                if(p_td.raw.comp == raw_sign_t.SG_SG_BIT) { //<<!!!!
                                                        min_bits++;
                                                }
                                                if (min_bits > p_td.raw.fieldlength) {
                                                        TTCN_EncDec_ErrorContext.error(error_type.ET_LEN_ERR, "There are insufficient bits to encode :
", p_td.name);
                                                        value = 0; // substitute with zero
                                                }
                                        }
```

- **Fix implemented.**

BUG 535860 - CREATION OF SOURCE-CODE ARCHIVE OF ONE OR MORE PROJECTS IN SINGLE FILE

- Creation of a feature similar to the "make archive" command, but with the ability to collect the needed source files from the referenced projects.
- The feature is accessible through right-clicking on a Titan project in the Project Explorer and selecting the "Create CLI archive" menu item.
- It will collect the files needed to compile the project via the command line tools and create a file named <ProjectName>_<Date>_<Time>.zip in the <ProjectRoot>/bin/backup directory.
- After unzipping the contents of this file in any directory the Makefile should be crated by running the
- "makefilegen ./*"
- command, or by modifying the original makefile, which is also included in the archive as "Makefile.orig"

BUG 534943 - OER DECODER IGNORES ASN.1 DEFAULT VALUES

- Fields of ASN.1 sequence types with default values are left unbound if they do not receive data while decoding. Instead, they should be set to their default values.

- Example:
- RecWithDefault ::= SEQUENCE
- {
- field1 INTEGER DEFAULT 1,
- field2 INTEGER
- }

- ...

- var octetstring enc := '000104'O;
- var RecWithDefault dec := dec_RecWithDefault(enc);
- // results in { field1 := <unbound>, field2 := 4 }
- // instead of { field1 := 1, field2 := 4 }

Fixed.

- OER regression tests updated to include default values in expected results

BUG 536084 - XSD2TTCN SOMETIMES INTERRUPTS COMMENTS

- Comments are sometimes interrupted when converting an XSD to TTCN-3 (the comment ends abruptly, and then continues in the next line as if it were a new comment).

- Example:
- regression_test/XML/XmlWorkflow/Tgc/confd.xsd (after the license header update to 2.0)

- ...
- the original definition of this type, it does not become "stuck" at
- its maximum or minimum value.)

- is converted into

- ...
- the original definition of this type, it does not become "stuck" at
- its maximum or minimum */
- /* value.) */

BUG 536084 - XSD2TTCN SOMETIMES INTERRUPTS COMMENTS

- Fixed this issue and another issue, where XML escape sequences (e.g. <) also interrupted comments.
- The expected TTCN-3 files that tested the previous, incorrect behavior have been corrected.

BUG 536130 - LICENSE UPGRADE TO EPL 2.0

BUG 536482 - INCORRECT CODE GENERATED FOR TYPE CONVERSION

- The compiler sometimes generates an extra block around type-conversion code. This causes a C++ compilation error, since the conversion result is declared inside this block and used after the block.

- Example:
- module test {
- type enumerated Enum { One, Two, Three };
- type record Rec1 {
- record of Enum f1
- }
- type record Rec2 {
- record of Enum f1
- }
- template Rec1 t_pard1(in template Rec1.f1 p) := { p };
- template Rec2 t_pard2(in template Rec1.f1 p) := { p };
- control {
- var template Rec1 x := t_pard1(t_pard2({ One, Two, Three }).f1); // C++ compilation error
- }
- }
- Fixed.
- The conversion result is now declared before the block.

- Regression tests added under regression_test/compileonly/typeCompat.

BUG 536713 - XSD2TTCN: INCORRECT CODE GENERATED FOR ELEMENT SUBSTITUTIONS FROM OTHER MODULES

- The XSD converter does not handle element substitutions correctly if some of the substituted elements are from other XSDs than the head element.

- For example, the attached XSD files generate the following union for the substitution group 'gml:_Surface':

```
type union Surface_group_1
{
        AbstractSurfaceType surface,
        ArcBand arcBand,
        Circle circle,
        Ellipse ellipse,
        Polygon polygon,
        Surface_group surface_1
}
with {
  variant "untagged";
  variant (surface) "name as '_Surface'";
  variant (surface) "form as qualified";
  variant (surface) "abstract";
  variant (arcBand) "name as capitalized";
  variant (circle) "name as capitalized";
  variant (ellipse) "name as capitalized";
  variant (polygon) "name as capitalized";
  variant (surface_1) "name as 'Surface'";
};
```

- This is generated in the same module as the head element (AbstractSurfaceType), with the target namespace prefix 'gml'. However, the next 4 fields are from a the top-level module (with the namespace prefix 'gs'), which is not imported here, thus causing a TTCN-3 compilation error. The namespace of these fields is also incorrect (they are supposed to be in the top-level module's namespace).

BUG 536713 - XSD2TTCN: INCORRECT CODE GENERATED FOR ELEMENT SUBSTITUTIONS FROM OTHER MODULES

- Proposed solution:
- - A new command line option should be added for xsd2ttcn, which causes all definitions to be generated into a single module, that contains a group for the definitions of every target namespace (so each group in this case would contain the contents that one module would contain, if this option were unset, minus the imports). This module will only import XSD.ttcn. (The helper modules, UsefulTtcn3Types.ttcn and XSD.ttcn, will still be generated.
- - Variant attributes should be generated for the fields of element substitution unions, to correct their namespaces (if necessary). This should be done whether the new command line option is set or not.
- - A warning should be issued if the new command line option is unset and the type of a substitution group element is not generated into the same module as the head element.

- See attachment for an example to reproduce the fault and its proposed solution (merged.ttcn).

BUG 537389 - IMPLEMENT NEW RAW CODING INSTRUCTION 'FORCEOMIT'

- A new RAW coding instruction is required, which sets lower-level optional fields to 'omit', when they are used from specific upper-level structures.

- Syntax: FORCEOMIT(<list of lower-level field identifiers>)

- Examples:
- type record InnerRec {
- integer opt1 optional,
- integer opt2 optional,
- integer opt3 optional,
- integer mand
- }

- // Note: decoding a value of type InnerRec alone does not force any of the
- // fields mentioned in the variants below to be omitted

- type record OuterRec1 {
- integer f1,
- InnerRec f2,
- integer f3
- }
- with {
- variant (f2) "FORCEOMIT(opt1)"
- }

BUG 537389 - IMPLEMENT NEW RAW CODING INSTRUCTION 'FORCEOMIT'

```
// Decoding ‘0102030405’O into a value of type OuterRec1 results in:
// {
//   f1 := 1,
//   f2 := { opt1 := omit, opt2 := 2, opt3 := 3, mand := 4 },
//   f3 := 5
// }

type record OuterRec2 {
  OuterRec1 f
}
with {
  variant (f) "FORCEOMIT(f2.opt2)"
}

// Decoding ‘01020304’O into a value of type OuterRec2 results in:
// {
//   f := {
//     f1 := 1,
//     f2 := { opt1 := omit, opt2 := omit, opt3 := 2, mand := 3 },
//     f3 := 4
//   }
// }
```

BUG 537389 - IMPLEMENT NEW RAW CODING INSTRUCTION 'FORCEOMIT'

```
type record OuterRec3 {
  OuterRec1 f1,
  OuterRec1 f2
}
with {
  variant (f1) "FORCEOMIT(f2.opt2, f2.opt3)"
  variant (f2) "FORCEOMIT(f2.opt2), FORCEOMIT(f2.opt3)"
}

// Decoding '010203040506'O into a value of type OuterRec3 results in:
// {
//   f1 := {
//     f1 := 1,
//     f2 := { opt1 := omit, opt2 := omit, opt3 := omit, mand := 2 },
//     f3 := 3
//   },
//   f2 := {
//     f1 := 4,
//     f2 := { opt1 := omit, opt2 := omit, opt3 := omit, mand := 5 },
//     f3 := 6
//   }
// }
```

- Change implemented.

- Regression tests added under regression_test/RAW/ForceOmit.
- Documented the new attribute in the reference guide.

BUG 537502 - CANONICAL OER ENCODING FOR DEFAULT VALUES

- Currently the OER encoder ignores a field's default value and always encodes its value into the resulting stream (and also indicates that the field is present). In Canonical OER, if a field is set to its default value, then it is encoded as if it were an omitted optional field (no data is added for the field, and its presence indicator is not set); thus making the result as short as possible.

- Example:
- Seq ::= SEQUENCE {
- field1 INTEGER,
- field2 INTEGER DEFAULT 1,
- field3 INTEGER DEFAULT 0
- }

- { field1 := 2, field2 := 1, field3 := 0 } should be encoded into '000102'O
- (current encoding: 'C0010201010100'O).

BUG 537502 - CANONICAL OER ENCODING FOR DEFAULT VALUES

- Change implemented and tests added.

- Note: The OER encoder now always omits fields whose values are equal to their default values (as per COER). This might cause backward incompatibility issues. If it does, then some setting should be added to switch between OER and COER.

BUG 537888 - INCORRECT OER ENCODING FOR POSITIVE AND RESTRICTED VALUE

- Values of the following type are encoded as if they could be negative values:
- "
- Psid ::= INTEGER (0..MAX)
- "

- However this is not the case. in OER encoding values that can not be negative, according to their type restriction, need a slightly different encoded octetstring.

BUG 537537 - PREPARE TITAN.CORE FOR RELEASE 6.4.1

- \- change the version numbers in the source code and the test

OER ENCODING: ENCODING OF EMPTY 'SEQUENCE OF' DATA TYPE GITHUB ISSUE#134

- Empty 'SEQUENCE OF' is currently encoded as '00'O octet string.
- Imho it has to be encoded as '0100'O octet string.

- From ISO/IEC 8825-7:2015 ch. 17.2:
- The quantity field shall be a non-negative integer value indicating the number of occurrences. This number shall be encoded as a length determinant followed by a variable-size unsigned number.

- Test data type:

- SequenceOfOctetString ::= SEQUENCE OF OCTET STRING (SIZE (2))

- Result from log:

- enc_OER_SequenceOfOctetString(): Encoding @TempA.SequenceOfOctetString: { }
- enc_OER_SequenceOfOctetString(): Stream after encoding: '00'O
- dec_OER_SequenceOfOctetString(): Stream before decoding: '0100'O
- dec_OER_SequenceOfOctetString(): Decoded @TempA.SequenceOfOctetString: { }

- Fixed

OER ENCODING: ENCODING OF EMPTY EXTENDABLE SEQUENCE GITHUB ISSUE #133

- Empty extendable sequence is currently encoded as an empty octet string.
- According to my understanding of ISO/IEC 8825-7:2015, it should be the '00'O one.

- Data type used for test:

- ...
- SequenceEmptyExtendable ::= SEQUENCE { ... }
- ...
- external function enc_OER_SequenceEmptyExtendable(in SequenceEmptyExtendable pdu) return octetstring with { extension "prototype(convert) encode(OER)" }

- Result from logs:

- Test case tc_OER_SequenceEmpty started.
- enc_OER_SequenceEmptyExtendable(): Encoding @TempA.SequenceEmptyExtendable: { }
- enc_OER_SequenceEmptyExtendable(): Stream after encoding: ‘’O

- Fixed

BUG 538465 - IMMUTABILITY CHECK CAUSES FATAL ERRORS

- Conformance test conformance_test/core_language_tests/positive_tests/16_functions_altsteps_testcases/1601_functions/160104_invoking_functions_from_specific_places/NegSem_160104_invoking_functions_from_specific_places_128.ttcn fails during compilation, with fatal error. This is caused by the immutability check on the actual parameter of function call 'f_test'.
- Other conformance tests in the same directory (no. 129, 171 and 172) also fail with the same fatal error, and likely have the same cause.
- Fixed the immutability check for 'out' and 'inout' actual parameters.

BUG 537888 - INCORRECT OER ENCODING FOR POSITIVE AND RESTRICTED VALUE

- Values of the folowing type are encoded as if they could be negative values:
- "
- Psid ::= INTEGER (0..MAX)
- "

- However this is not the case. in OER encoding values that can not be negative, according to their type restriction, need a slightly different encoded octetstring.

BUG 539726 - PTC VERDICT REASON IS NOT TRANSMITTED TO THE MTC IF THE COMPONENT IS 'ALIVE'

- If the PTC is created with the 'alive' option, then its verdict reason is lost, when the MTC calculates the final verdict.

- Example code:
- module MyExample
- {
- type component PCO_CT {}
- type component MTCType{
- var PCO_CT pcoComponent;
- }
- function f_helloWorld() runs on PCO_CT{
- setverdict(pass,"here is my reason on PCO_CT");
- }
- testcase tc_HelloW() runs on MTCType system PCO_CT
- {
- pcoComponent := PCO_CT.create("MyPCOComp") alive;
- pcoComponent.start(f_helloWorld());
- pcoComponent.done;
- }
- control
- {
- execute(tc_HelloW());
- }
- }
- Got:
- Test case tc_HelloW finished. Verdict: pass
- Expected:
- Test case tc_HelloW finished. Verdict: pass reason: here is my reason on PCO_CT

BUG 539726 - PTC VERDICT REASON IS NOT TRANSMITTED TO THE MTC IF THE COMPONENT IS 'ALIVE'

- Fixed.
- The verdict reason is now properly transmitted to the Main Controller when the PTC is killed (previously an empty reason was transmitted, which had overwritten the reason set when the behavior function ended).

BUG 539956 - XER DEC: INVALID DECODING OF ANYELEMENT IN RECORD OF RECORD

- The XER decoder fails to decode a universal charstring with the 'anyElement' attribute if it is the first (or only) field of a record of record.

- Example:

```
module mini {
type record Main
{
                record of AnyElem anyElem_list
}
with {
  variant "name as uncapitalized";
  variant (anyElem_list) "untagged";
};

type record AnyElem
{
                universal charstring elem
}
with {
  variant "untagged";
  variant (elem) "anyElement";
};
control {
  var universal charstring data := "<main> <something/> </main>";
  var Main x;
  action(decvalue_unichar(data, x), x); // Unprocessed XML tag `something'
}
}
with {
  encode "XML";
}
```

BUG 539956 - XER DEC: INVALID DECODING OF ANYELEMENT IN RECORD OF RECORD

- Fixed.
- Regression tests added to regression_test/XML/EXER-whitepaper/AnyElement.ttcnpp.
- The mentioned fix was reverted, because it caused a regression test to fail in RT2, and it caused a different decoding error in the user project.
- A new fix was implemented, which handles the 'anyElement' in the record type's generated code, instead of the UNIVERSAL_CHARSTRING class.

- Note: There are still cases, in which the 'anyElement' attribute is not handled properly in XML decoding (when they are even further embedded in one-element records/unions). Fixing these would probably require moving the 'anyElement' attribute checks from the code generator (i.e. from compile-time) to the generated code (i.e. runtime), so they can go deeper into structures. This could reduce the performance of XML decoding significantly.

BUG 540051 - NEW JSON ATTRIBUTE 'AS MAP'

- The attribute can be set for record of/set of types, whose element types are records/sets with 2 fields. The first field has to be a non-optional universal charstring. Values of affected types are encoded as a JSON object (instead of a JSON array), where each key-value pair is created from the 2 fields of each element in the record of/set of (the universal charstring field's value will be the key, and the 2nd field's JSON encoding will be the value).

- Example:
- type record MapItem {
- universal charstring key,
- integer value_
- }

- type set of MapItem Map
- with { variant "as map" }

- ...

- var Map x := { { "one", 1 }, { "two", 2 }, { "three", 3 } };
- var universal charstring str := encvalue_unichar(x);

- Result: {"one":1,"two":2,"three":3}

BUG 540262 - ATTRIBUTE 'TEXT ... AS ...' FOR JSON

- Implement the XML attribute 'text ... as ...' for JSON, too.

- For example:

```
type enumerated EnumNumber { One, Two, Three }
with {
  variant "text 'One' as '1'";
  variant "text 'Two' as '2'";
  variant "text 'Three' as '3'";
}
type record of EnumNumber EnumNumberList;

const EnumNumberList c_numbers := { One, Two, Three };
// is encoded into: [ "1", "2", "3" ]
```

- Limitations compared to the XML version:
  - 'text all as ...' doesn't work, only individual enum values can be changed;
  - the XML-specific keywords (such as 'uncapitalized', etc.) also don't work, the new text must always be specified.

BUG 540607 - INVALID DECODING OF XER LIST WITH LEADING WHITESPACE

- The XML decoder fails to decode record ofs/set ofs with the 'list' encoding instruction, if there are whitespaces between the end of the XML tag and the beginning of the actual list in the XML document. This usually results in the first list item being successfully decoded, while the rest are ignored.

- For example:
- type record of universal charstring List
- with {
- encode "XML";
- variant "name as uncapitalized";
- variant "list";
- };

- control {
- var List x;
- var universal charstring y := "<list>
- a
- b
- c
- </list>";
- action(decvalue_unichar(y, x), x); // 0{ x := "a" }
- }

- Fixed (the string tokenizer did not account for white spaces at the beginning of the string).

- Regression test case added to regression_test/XML/EXER-whitepaper/List.ttcnpp.

GITHUB PULL REQUEST #139

- OER encoding of big INTEGER #139

MAKEFILEGEN: ADOPT FOR THE DEBIAN PACKAGING

› https://github.com/eclipse/titan.core/commit/43e665c4ff9187031179bd2c74327378ba48595d

› The Debian and Ubuntu packages install the
› - headers into /usr/include/titan
› - libraries into /usr/lib/titan

› So the generated Makefile tries to detect the presence of these
› directories **if the TTCN3_DIR is not set**, and adjust the CPPFLAGS and the
› linker command

› Signed-off-by: Gabor Szalai <gabor.szalai@ericsson.com>

BUG 541100 - COMPILER CRASH DURING CODE GENERATION FOR TYPE CONVERSION

- The following code crashes the compiler with a fatal error, while attempting to generate code for a value that needs type conversion.

- Code:
- module mini {

- type record StatusLine {
- charstring sipVersion,
- integer statusCode,
- charstring reasonPhrase
- }
- type record StatusLine_2xx {
- charstring sipVersion,
- integer statusCode (200 .. 299) ,
- charstring reasonPhrase
- }
- type record Main {
- StatusLine_2xx field
- }
- template (value) Main t(template(value) StatusLine_2xx p) := { p };
- const StatusLine c := { "a", 1, "b" };
- control {
- action(t(c));
- }
- }
- Fixed & tests added.

BUG 539514 - REAL-TIME TESTING IN TITAN

- Implement the real-time testing tools described in chapter 5 of the TTCN-3 standard extension document 'TTCN-3 Performance and Real-time Testing' (ES 202 782).

- Implemented operation 'now', which queries the test system clock.

BUG 539514 - REAL-TIME TESTING IN TITAN

- Implemented the timestamp redirect for incoming port operations ('receive', 'getcall', 'getreply', 'catch', and their 'check' variants).
- Limitation: Internal ports do not set their timestamps automatically (in these cases the redirect returns an unbound float). Currently the redirects can only be used on user ports.
- Tests added to regression_test/realtime (still missing tests for timestamp redirects in 'any_from' constructs and in translation ports) and function_test/Semantic_Analyser/realtime.
- Note: The newly added keywords ('now', 'realtime' and 'timestamp') might cause backward incompatibilities. Specifically 'timestamp' seems to be used in a lot of XSD's that are converted to TTCN-3.

BUG 539514 - REAL-TIME TESTING IN TITAN

- Added compiler option '-I', which enables the real-time testing features mentioned here. The features are disabled by default, and the new keywords ('now', 'realtime' and 'timestamp') can be used as identifiers again (for backward compatibility).

- Also added makefilegen option '-i', which activates this option for the compiler, and the makefile setting 'enableRealtimeTesting' in the TPD, which does the same thing.

BUG 539514 - REAL-TIME TESTING IN TITAN

- Implemented the timestamp redirect for outgoing port operations ('send', 'call', 'reply' and 'raise').
- Limitation: Internal ports do not set their timestamps automatically (in these cases the redirect returns an unbound float). Currently the redirects can only be used on user ports.
- Tests added to regression_test/realtime (still missing tests for timestamp redirects in translation ports) and function_test/Semantic_Analyser/realtime.

GITHUB:LOG RESULT WHEN TC FINISHES WITH VERDICT INCONC #143

BUGZILLA JUNE

Bug ID	Product	Component	Assignee	Status	Resolution	Summary	Changed
528094	Titan	Plug-ins	arpad.lovassy@semcon.com	CLOSED	FIXED	Numeric value of enumerated can be defined as any type convertible to integer	5/24/2018 10:00
535860	Titan	Plug-ins	titan-inbox@eclipse.org	RESOLVED	FIXED	Creation of source-code archive of one or more projects in single file	6/13/2018 7:54
535558	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	XML enc/dec: any element namespace restrictions should not affect sub-elements	6/14/2018 4:16
535441	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Fatal error when using extended component as actual parameter	6/14/2018 4:16
521417	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	ASN.1 BITSTRING subtyping false error	6/14/2018 4:17
535834	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	incorrect decoding is not indicated in a corner case	6/15/2018 11:26
532595	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	RAW encoder should not accept integer numbers out of range	6/15/2018 11:26
534943	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	OER decoder ignores ASN.1 default values	6/18/2018 9:43
535729	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Fault in re-arrangement of template default parameters (RT2 only)	6/18/2018 11:46
535800	Titan	Plug-ins	arpad.lovassy@semcon.com	CLOSED	FIXED	Decmatch should be accepted by the syntax checker according to the standard	6/21/2018 6:26
536084	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	xsd2ttcn sometimes interrupts comments	6/29/2018 9:35

BUGZILLA JULY-AUGUST

Bug ID	Product	Component	Assignee	Status	Resolution	Summary	Changed
536582	Titan	Plug-ins	jeno.balasko@ericsson.com	RESOLVED	FIXED	Spaces are removed from free strings of files .TITAN_properties and .tpd	7/2/2018 12:52
537537	Titan	Core	jeno.balasko@ericsson.com	RESOLVED	FIXED	Prepare titan.core for release 6.4.1	7/31/2018 11:01
537502	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Cannonical OER encoding for default values	8/31/2018 11:10

BUGZILLA SEPTEMBER

Bug ID	Product	Component	Assignee	Status	Resolution	Summary	Changed
538482	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Type descriptor generation problems caused by OER fix	9/4/2018 9:36
537888	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	incorrect OER encoding for positive and restricted value	9/4/2018 9:41
538465	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Immutability check causes fatal errors	9/4/2018 9:56
537389	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Implement new RAW coding instruction 'FORCEOMIT'	9/4/2018 10:19
536482	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Incorrect code generated for type conversion	9/4/2018 10:25
536713	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	xsd2ttcn: incorrect code generated for element substitutions from other modules	9/4/2018 10:36

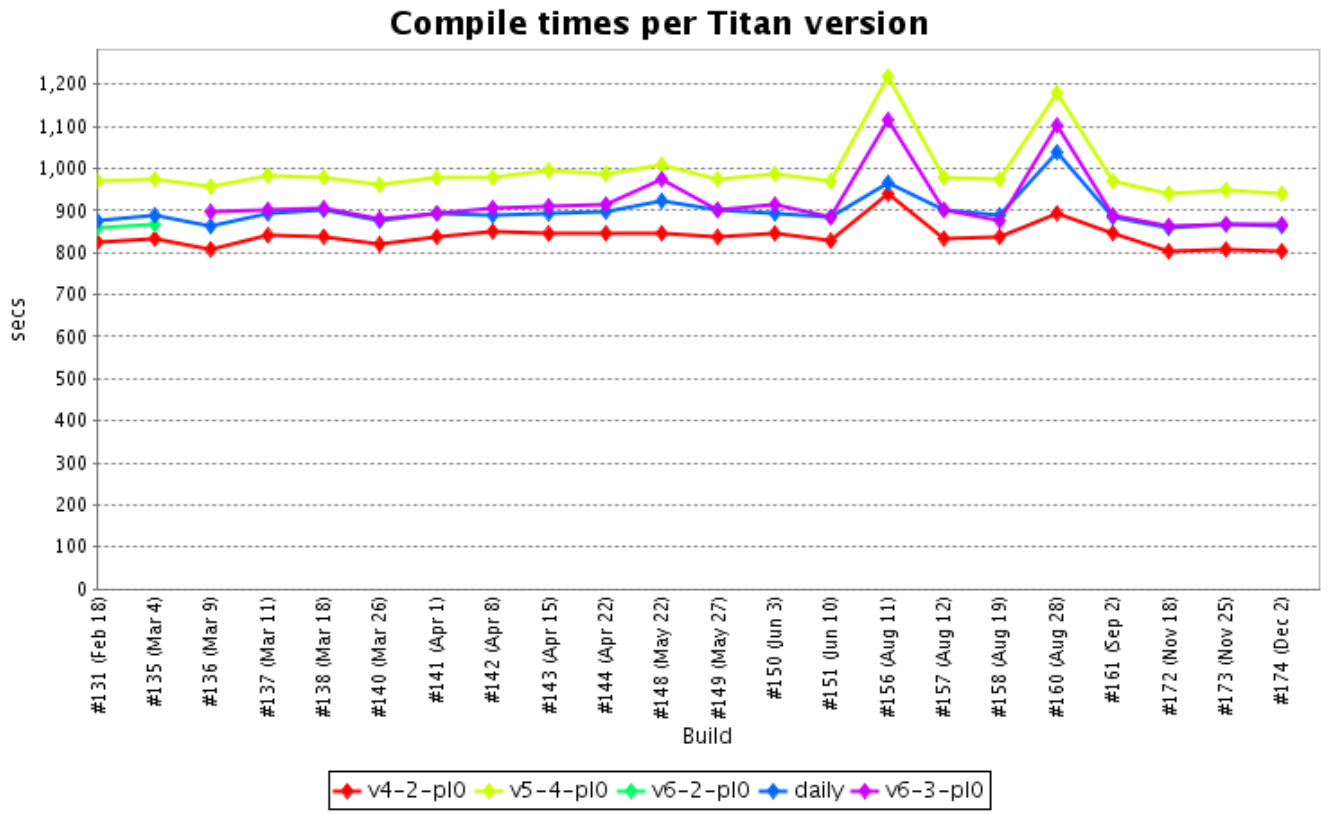
BUGZILLA OCTOBER

536132	Titan	Plug-ins	jeno.balasko@ericsson.com	CLOSED	FIXED	License upgrade to EPL 2.0 in plugins	10/2/2018 9:34
537537	Titan	Core	jeno.balasko@ericsson.com	CLOSED	FIXED	Prepare titan.core for release 6.4.1	10/2/2018 9:36
539726	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	PTC verdict reason is not transmitted to the MTC if the component is 'alive'	10/2/2018 9:43
539956	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	XER dec: invalid decoding of anyElement in record of record	10/16/2018 4:48
540217	Titan	Plug-ins	jeno.balasko@ericsson.com	CLOSED	FIXED	Template module parameter should not accepted in value comparison	10/18/2018 3:52
540051	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	New JSON attribute 'as map'	10/18/2018 11:35
540262	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Attribute 'text ... as ...' for JSON	10/19/2018 7:09
540607	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Invalid decoding of XER list with leading whitespace	10/30/2018 11:42

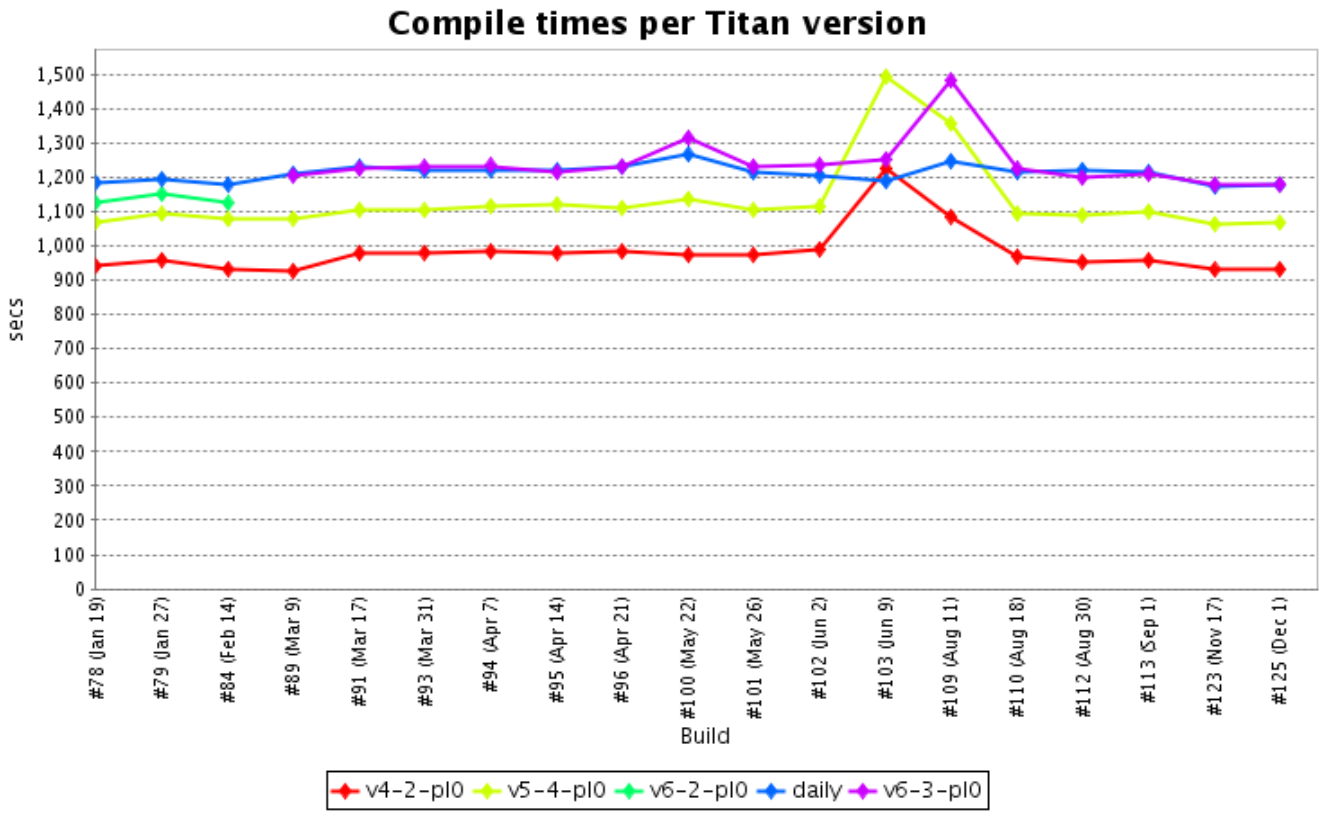
BUGZILLA NOVEMBER

541100	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Compiler crash during code generation for type conversion	11/14/2018 5:39
539514	Titan	Core	botond.baranyi@ericsson.com	ASSIGNED	---	Real-time testing in TITAN	11/28/2018 4:46
541634	Titan	Core	jeno.balasko@ericsson.com	RESOLVED	FIXED	Missing html pages in the on-line help	11/28/2018 6:58

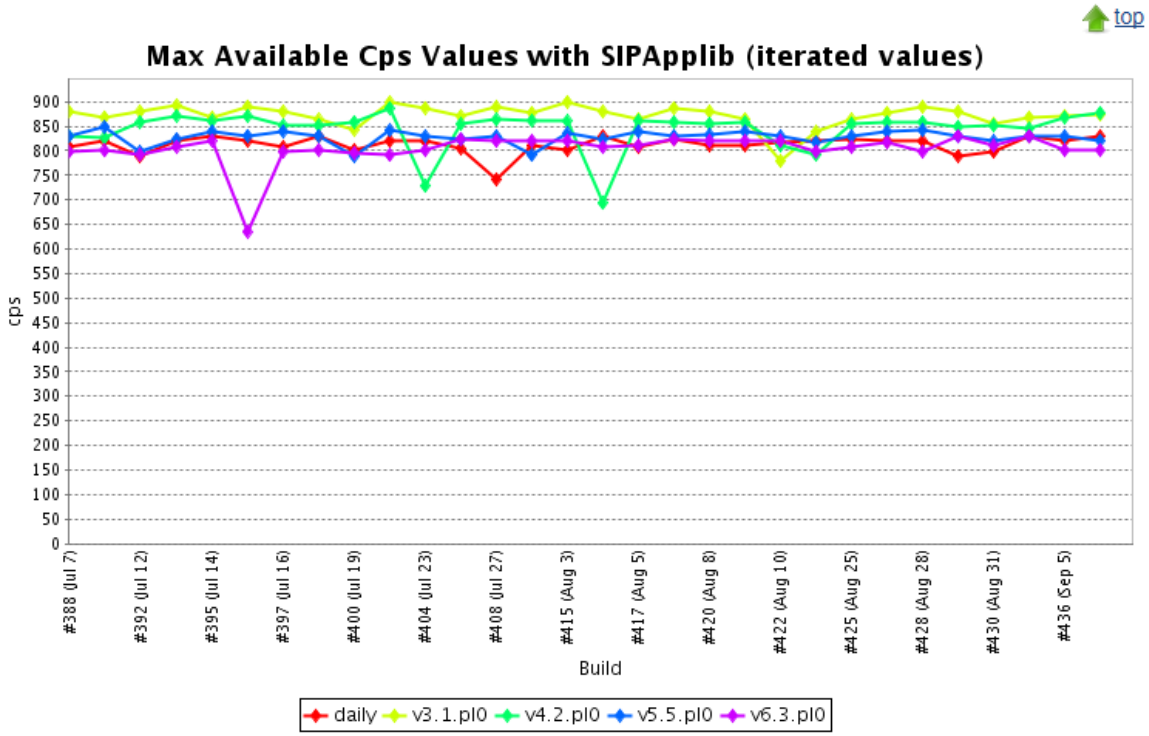
COMPILE TIMES RT1



COMPILE TIMES RT2



EXECUTION PERFORMANCE RT1



ERICSSON