ERICSSON

TITAN 6.6.0 CHANGE LOG (COVERING 6.5.2 & 6.5.1)

2019.05.16

I.SUMMARY

NEW ECLIPSE JAVA CODE GENERATION PLUG-IN

- https://github.com/eclipse/titan.core/blob/master/usrguide/java_referenceguide/JavaReferenceGuide.adoc

- See “Tutorial for the Java code generator and executor of the TITAN Eclipse”:
- https://github.com/eclipse/titan.core/blob/master/usrguide/JavaCodegenExecTutorial.pdf

NEW ECLIPSE JAVA CODE GENERATION PLUG-IN

- Note: due to the need to sync the C++ and Java side, a new restriction had to be introduced:
- **The version of the Eclipse plug-ins has to coincide with the version of the Titan core compiler and executor, else the plug-ins will not work.**
- This restriction is not present in versions 6.5.0 and earlier.

NEW ECLIPSE REFACTORING PLUG-IN

- https://github.com/eclipse/titan.EclipsePlug-ins/blob/master/org.eclipse.titanium.refactoring/docs/Titanium_Refactoring_Description/Titanium_Refactoring_Description.adoc

NEW SWITCHES

- Bug 544861 - Printing symbolic version in 'compiler -v’
- Bug 545308 - Legitimize compiler option '-0’
- Bug 540052 - Get list of TSPs from TTCN binary: New command line option '-p' implemented for TTCN binaries (both in single mode and parallel mode), which lists all module parameters.

NEW FEATURES

- Bug 539514 - Support for real-time testing in TITAN
- Bug 529443 - Extend 'default' JSON attribute for structured types
- Bug 540052 - Get list of TSPs from TTCN binary
- Bug 544861 - Printing symbolic version in 'compiler -v'
- Bug 545308 - Legitimize compiler option '-0'
- Bug 546286 - Display warnings for 'map param' and 'unmap param'- Unlike the name suggests, this is a complete implementation of the compiler side of **map param/unmap param**
- Bug 547318 - Colorize compiler error/warning messages

BUGFIXES

- Bug 538482 - Type descriptor generation problems caused by OER fix
- Bug 541748 - xsd2ttcn: Name clashes and references not handled when using single module
- Bug 543155 - New error behavior option for data remaining after decoding
- Bug 542610 - Translation ports across multiple Eclipse projects- documentation
- Bug 539612 - Ubuntu 18.04, gcc 7, c++11 compatibility
- Bug 517843 - Support for multiple encodings - Updated the syntax analysis of '<port>.setencode' to allow subreferences
- Bug 537885 - Invalid initialization of reference (c++ error message); @fuzzy testcase parameter with default value.
- Bug 537979 - @fuzzy as formal parameter: unspecified limitation of feature
- Bug 537922 - Incorrect order in record generated by xsd2ttcn
- Bug 539393 - XER enc: encvalue uses incorrect type descriptor
- Bug 537328 - Bad assignment for records with optional fields
- Bug 544632 - Duplicate identifier usage causes fatal error
- Bug 544828 - Clashing codecs cause code generation error with legacy codec handling
- Bug 544855 - isvalue might throw DTE
- Bug 544978 - error when forced seof generation is set
- Bug 544608 - report the ASN.1 feature "extensibility implied" as not yet supported
- Bug 541693 - Constraints on ASN.1 integers with named values ignored by OER encoder
- Bug 545802 - log(omit) causes compiler crash
- Bug 546800 - Define function type which parameters with timer would get compiler error
- Bug 547307 - Backward incompatibility caused by octetstring alignment change in RAW codec

II.BUG DETAILS

BUG 539514 - REAL-TIME TESTING IN TITAN

- Continued from 6.5.0:
- Added default values to generated port class member functions 'send', 'call', 'reply' and 'raise' for backward compatibility (so they can be called from external functions without the timestamp redirect parameter).
- Regression test added, too.

BUG 538482 - TYPE DESCRIPTOR GENERATION PROBLEMS CAUSED BY OER FIX

- A fix to OER problem https://bugs.eclipse.org/bugs/show_bug.cgi?id=537888 (which has now been removed) caused type descriptors for certain ASN.1 types to be generated incorrectly, when using XER in ASN.1 (compiler option '-a').
- The error occurred in test regression_test/compileonly/openTypeXER.
- The OER fix has been re-added, and the type descriptor fault has been fixed.

BUG 541748 - XSD2TTCN: NAME CLASHES AND REFERENCES NOT HANDLED WHEN USING SINGLE MODULE

- The following problems occur when generating a single module, using the command line option '-o':

- - Name clashes: two types with the same name in different XSDs cause name clash errors in the generated TTCN-3 file (normally these would be generated into different modules and not cause errors).
- Solution: suffix the types (e.g. with _1, _2, etc.) and give them 'name as' attributes.

- - References to other modules: references to values from other namespaces (e.g. default values) are still prefixed with their module names.
- Solution: remove the prefixes, as they are in the same module.

- For examples see attached XSD files.
- Solutions implemented.
- Regression test added under regression_test/XML/XmlWorkflow
- Tests have passed.
- Improved this fix to work for name clashes across 3 or more XML namespaces.

- Regression test updated.

BUG 543155 - NEW ERROR BEHAVIOR OPTION FOR DATA REMAINING AFTER DECODING

- Currently, if there is data left in the buffer after decoding succeeds, the decoder function always displays a warning about it.
- A new option should be added in the 'errorbehavior' extension of decoder functions to change this warning to error or to ignore it.
- Implemented error behavior 'EXTRA_DATA', which governs the mentioned warning
  Possible backward incompatibility: setting 'errorbehavior' to 'ALL:ERROR' now also set this to error, which would cause code that previously only displayed a warning to now display a DTE.

BUG 542610 - TRANSLATION PORTS ACROSS MULTIPLE ECLIPSE PROJECTS

- The generated code for provider ports (i.e. ports in the 'map to' clauses of translation ports) changes, depending on how many translation ports are mapped to them.
- Because of this the provider port's module and all modules containing translation ports that are mapped to the provider port must be in the same compiler command. If the provider port is compiled first on its own (i.e. without the modules that contain translation ports it is mapped to), and the compiled C++ code is used in a different project, where the translation ports are (such as with central storage, and with project references in the Eclipse plug-ins), then the generated C++ class for the provider port won't contain all members and methods necessary for the port translation (causing C++ compilation errors in the second project).

- This limitation has been documented in the reference guide.

- A possible solution for this issue should be investigated.

- Created attachment 276873 [details]
- Example

BUG 542610 - TRANSLATION PORTS ACROSS MULTIPLE ECLIPSE PROJECTS

- Proposed solution:

- The classes generated for all ports with the 'provider' attribute would contain all necesary code to handle port translation (with any translation port).
- This would include the following changes:
- - a port pointer array (PORT**) and its size as new members for the provider port's class, which would be used in the same way as the current port-type-specific pointer arrays (p_0, p_1, etc. and n_0, n_1, etc.);
- - the member functions 'add_port' and 'remove_port' would be generated for all provider port classes and would add or remove ports from the new pointer array (without dynamic casting, since the actual port types are unknown);
- - the member functions 'reset_port_variables' and the 'outgoing_public_send/call/reply/raise' functions for all message/signature types would also be generated (their contents would be unchanged from the ones generated for current provider ports);
- - the member functions 'incoming_message/call/reply/exception' would contain the 'for' cycle that goes through the mapped translation ports (from current provider port code), except now it would go through all ports, and call a new function in the port base class that calls the translation port's 'incoming_message/call/reply/exception' member function (since these are not part of the base port class).

- The base port class (PORT) would have the following new function:

- virtual boolean incoming_message_handler(const void* message_ptr, const char* message_type, component sender_component, const FLOAT& timestamp);

- This would be implemented by all translation ports, and it would call the appropriate 'incoming_message' function for the type indicated by the parameter 'message_type'. There would either be four of these functions in total (3 more for calls, replies and exceptions), or it would have an extra parameter that indicates the operation type.

- Performance:
- - in case of translation ports there would be minor performance losses (extra functions being called, more generated code, etc.);
- - if the provider port is used in normal mode, or in a dual-faced port, then there would be a considerable amount of extra generated code, and minor performance losses during runtime.

BUG 542610 - TRANSLATION PORTS ACROSS MULTIPLE ECLIPSE PROJECTS

- Created attachment 276938 [details]
- Modified C++ files with the proposed solution

- These files contain the extra code that would be generated into A.cc, A.hh, B.cc and B.hh in the previously attached example (modifications are marked with lots of '/'-es).

- This still needs the addition of the incoming_message_handler function in PORT.hh and its base implementation (which just returns FALSE) in PORT.cc in the runtime library.
- Proposed solution implemented.

- Regression tests added under regression_test/portTranslationCentralStorage.

BUG 539612 - UBUNTU 18.04, GCC 7, C++11 COMPATIBILITY

- The ttcn3float.hh is not compatible with the c++11, gcc 7 used by Ubuntu 18.04
- In c++11 the signbit() is not a macro any more, so the
- #ifndef signbit is true and the ttcn3float.hh redeclares the signbit as a macro.
- It doesn't cause any problem until something other than struct ttcn3float tries to use the signbit().
- For example an #include <complex> after the #include <TTCN3.hh>
- Either protect the signbit checking with
- #if __cplusplus < 201103L // c++11 -> __cplusplus == 201103L
- #ifndef signbit
- ....
- #endif // def signbit
- #endif // __cplusplus < 201103L
- Or remove it.
- Implemented the proposed extra C++ version check.

BUG 517843 - SUPPORT FOR MULTIPLE ENCODINGS

- Continued

- Updated the syntax analysis of '<port>.setencode' to allow subreferences (i.e. port array indexes). This form of 'setencode' is still not supported, but the compiler now displays a more appropriate error message for it.

BUG 537885 - INVALID INITIALIZATION OF REFERENCE (C++ ERROR MESSAGE); @FUZZY TESTCASE PARAMETER WITH DEFAULT VALUE.

- TTCN-3 code:

```
module M {
   type component C {
      var integer CV_I := 0;
   }

   function F() runs on C return integer {
      CV_I := CV_I + 1;
      return CV_I;
   }

   testcase T (in @fuzzy integer FFP_I := 4) runs on C {
      log(FFP_I); log(FFP_I); log(FFP_I); log(FFP_I); log(FFP_I);
   }

   control {
      execute(T(3));
   }
}
```

- make result:

```
g++  -c -DLINUX -I/home/aron/titan.core/Install/include -DMEMORY_DEBUG -Wall   -g -o M.o M.cc
M.cc: In function 'verdicttype M::testcase_T_defparams(boolean, double)':
M.cc:134:21: error: invalid initialization of reference of type 'Lazy_Fuzzy_Expr<INTEGER>&' from expression of type 'const INTEGER'
  return testcase_T(T_FFP__I_defval, has_timer, timer_value);
                    ^~~~~~~~~~~~~~~
M.cc:70:13: note: in passing argument 1 of 'verdicttype M::testcase_T(Lazy_Fuzzy_Expr<INTEGER>&, boolean, double)'
 verdicttype testcase_T(Lazy_Fuzzy_Expr<INTEGER>& FFP__I, boolean has_timer, double timer_value)
             ^~~~~~~~~
Makefile:140: recipe for target 'M.o' failed
make: *** [M.o] Error 1
```

BUG 537885 - INVALID INITIALIZATION OF REFERENCE (C++ ERROR MESSAGE); @FUZZY TESTCASE PARAMETER WITH DEFAULT VALUE.

- Fixed.
- The lazy/fuzzy classes and objects are now generated for testcase parameter default values, too.
- Tests have passed.

BUG 537979 - @FUZZY AS FORMAL PARAMETER: UNSPECIFIED LIMITATION OF FEATURE

- Created attachment 275421 [details]
- testset for fuzzy

- The attached fuzzy.script file serve some example of the feature usage and it's behavior. These tests try feature of @fuzzy as formal parameter of function/testcase/altstep with or without default parameter.

- Functions are working correctly with fuzzy parameter which get from caller, but these getting unexpected error from compiler if we use with default parameter (the expected behavior of compiler and executor are same).

- Testcases can not working with fuzzy parameters, but the compiler don't warning.

- Altsteps work like functions except the "Segmentation fault" (only with activate()).

- See also: https://bugs.eclipse.org/bugs/show_bug.cgi?id=537885

BUG 537979 - @FUZZY AS FORMAL PARAMETER: UNSPECIFIED LIMITATION OF FEATURE

- Fixed the mentioned internal DTE and segmentation fault by adding a compiler-time check that disallows altsteps with lazy/fuzzy parameters in the 'activate' operation.

- There are two other compiler errors in the mentioned tests:
- - calling a function with a runs-on clause from the control part (when testing testcases with fuzzy parameters) - this limitation is in compliance with the TTCN-3 standard; using the return value of a function without a runs-on clause as a testcase's fuzzy parameter works fine;
- - setting the default value of a function/testcase/altstep parameter to the return value of a function call - this is a general limitation in TITAN (though it might be undocumented).

BUG 537922 - INCORRECT ORDER IN RECORD GENERATED BY XSD2TTCN

- When in an xsd an substitution group is referenced, the right order is not preserved (see attached XSD)

- This

```
<xs:element name="mapping">
   <xs:complexType>
     <xs:sequence>
       <xs:element minOccurs="0" maxOccurs="unbounded" ref="ns1:displayName"/>
       <xs:group ref="ns1:serviceGroup"/>
   <!-- xs:sequence>
     <xs:element minOccurs="0" ref="ns1:service"/>
   </xs:sequence -->

       <xs:choice minOccurs="0">
         <xs:group ref="ns1:serviceBoundary"/>
         <xs:element ref="ns1:serviceBoundaryReference"/>
       </xs:choice>
       <xs:element minOccurs="0" maxOccurs="unbounded" ref="ns1:uri"/>
       <xs:element minOccurs="0" ref="ns1:serviceNumber"/>
       <xs:group ref="ns1:extensionPoint"/>
     </xs:sequence>
     <xs:attributeGroup ref="ns1:expires"/>
     <xs:attribute name="lastUpdated" use="required" type="xs:dateTime"/>
     <xs:attributeGroup ref="ns1:source"/>
     <xs:attribute name="sourceId" use="required" type="xs:token"/>
     <xs:attributeGroup ref="ns1:message"/>
   </xs:complexType>
 </xs:element>
```

BUG 537922 - INCORRECT ORDER IN RECORD GENERATED BY XSD2TTCN

- will lead to this equivalent TTCN-3 code:

```
type record Mapping
{
        Lang lang optional,
        XSD.AnySimpleType expires,
        XSD.DateTime lastUpdated,
        XSD.Token message_ optional,
        AppUniqueString source,
        XSD.Token sourceId,
        record of DisplayName displayName_list,
        union {
                ServiceBoundary_1 serviceBoundary,
                ServiceBoundaryReference serviceBoundaryReference
        } choice optional,
        record of Uri uri_list,
        ServiceNumber serviceNumber optional,
        Service service optional,
        record of NotLost notLost_list
}
with {
  variant "name as uncapitalized";
  :
};
```

- where service is declared after serviceBoundary violating the order.

BUG 537922 - INCORRECT ORDER IN RECORD GENERATED BY XSD2TTCN

- If the substitution group is replaced with the element declaration the order is preserved:

```
<xs:element name="mapping">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="ns1:displayName"/>
      <!-- xs:group ref="ns1:serviceGroup"/ -->
  <xs:sequence>
    <xs:element minOccurs="0" ref="ns1:service"/>
  </xs:sequence>

      <xs:choice minOccurs="0">
        <xs:group ref="ns1:serviceBoundary"/>
        <xs:element ref="ns1:serviceBoundaryReference"/>
      </xs:choice>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="ns1:uri"/>
      <xs:element minOccurs="0" ref="ns1:serviceNumber"/>
      <xs:group ref="ns1:extensionPoint"/>
    </xs:sequence>
    <xs:attributeGroup ref="ns1:expires"/>
    <xs:attribute name="lastUpdated" use="required" type="xs:dateTime"/>
    <xs:attributeGroup ref="ns1:source"/>
    <xs:attribute name="sourceId" use="required" type="xs:token"/>
    <xs:attributeGroup ref="ns1:message"/>
  </xs:complexType>
</xs:element>\>
```

BUG 537922 - INCORRECT ORDER IN RECORD GENERATED BY XSD2TTCN

```
type record Mapping
{
        Lang lang optional,
        XSD.AnySimpleType expires,
        XSD.DateTime lastUpdated,
        XSD.Token message_ optional,
        AppUniqueString source,
        XSD.Token sourceId,
        record of DisplayName displayName_list,
        Service service optional,
        union {
                ServiceBoundary_1 serviceBoundary,
                ServiceBoundaryReference serviceBoundaryReference
        } choice optional,
        record of Uri uri_list,
        ServiceNumber serviceNumber optional,
        record of NotLost notLost_list
}
with {
  variant "name as uncapitalized";
  variant "element";
   :
};
```

- Fixed.
- The contents of referenced groups are now properly generated at the group's location instead of at the end of the record.

- Regression tests added under regression_test/XML/XmlWorkflow.

BUG 529443 - EXTEND 'DEFAULT' JSON ATTRIBUTE FOR STRUCTURED TYPES

- According to the TTCN-3 standard, the JSON attribute 'default' can be used on record/set fields of any type (including structured types), and its parameter (in brackets) is a TTCN-3 value, which is assigned to the field (during decoding) if it doesn't appear in the JSON document.
- In TITAN this attribute is only usable on fields of built-in types, and the attribute's parameter is a JSON value that is decoded by the field if it doesn't appear in the JSON document.
- As a first step, the attribute should be extended in TITAN to work for empty structures (i.e. {}), as this would be enough for most use cases. The syntax of the attribute's parameter should also be changed to a TTCN-3 value (as in the standard) from the current JSON value syntax.
- Examples:
- const charstring json_data := "{\"field1\":2}";
- type record Rec1 {
- integer field1,
- record of integer field2
- } with { encode "JSON"; variant field2 ”default({})” }
- // json_data is decoded into { field1 := 2, field2 := omit }
- type record Rec2 {
- integer field1,
- record of integer field2 optional
- } with { encode "JSON"; variant field2 ”default({})” }
- // json_data is decoded into { field1 := 2, field2 := omit }

BUG 529443 - EXTEND 'DEFAULT' JSON ATTRIBUTE FOR STRUCTURED TYPES

- type record Rec3 {
- integer field1,
- record of integer field2 optional
- } with { encode "JSON" }

- // json_data is decoded into { field1 := 2, field2 := omit }

- type record Rec4 {
- integer field1,
- record of integer field2
- } with { encode "JSON" }

- // json_data causes a decoding error
- Change implemented: the attribute 'default' was extended to work for empty structures.

- Regression tests added to regression_test/json/AttributeTestcases.ttcn.

BUG 539393 - XER ENC: ENCVALUE USES INCORRECT TYPE DESCRIPTOR

- In the following example, when using legacy codec handling, the compiler incorrectly generates the referenced type's descriptor for XER encoding with 'encvalue'. If the same XER encoding is done through an external function, then it is generated correctly.

```
module test {

import from XSD all; // generated by xsd2ttcn
import from UsefulTtcn3Types all; // generated by xsd2ttcn

type charstring NCNameList;
type charstring Id;
type charstring Pos;

type PointType Point
with {
  variant "element";
};
type record PointType
{
  NCNameList axisLabels optional,
  XSD.String gid optional,
  Id id optional,
  XSD.PositiveInteger srsDimension optional,
  XSD.AnyURI srsName optional,
  NCNameList uomLabels optional,
  Pos pos
}
with {
  variant (axisLabels) "attribute";
  variant (gid) "attribute";
  variant (id) "attribute";
  variant (srsDimension) "attribute";
  variant (srsName) "attribute";
  variant (uomLabels) "attribute";
};
```

BUG 539393 - XER ENC: ENCVALUE USES INCORRECT TYPE DESCRIPTOR

```
external function f_enc(in Point x) return octetstring
  with { extension "prototype(convert) encode(XER:XER_EXTENDED)" };

control {
  var Point x := { omit, omit, "point1", omit, "urn:ogc:def:crs:EPSG::4326", omit, "48.215388 16.290300" };
  action(oct2char(f_enc(x))); // encoded as XML tag <gml:Point> (correct)
  action(oct2char(bit2oct(encvalue(x)))); // encoded as XML tag <gml:PointType> (incorrect)
}

}
with {
  encode "XML";
  variant "namespace as 'http://www.opengis.net/gml' prefix 'gml'";
  variant "controlNamespace 'http://www.w3.org/2001/XMLSchema-instance' prefix 'xsi'";
  variant "elementFormQualified";
}
```

- **Limitation documented.**

BUG 537328 - BAD ASSIGNMENT FOR RECORDS WITH OPTIONAL FIELDS

- The 'empty'/'{}' assignment is not correct for record only with optional fields: the value is always unbound instead of being bound.

- TITAN Product number: CRL 113 200/5 R5A

- Example:
- ------
- ...
- type record r
- {
- integer f1 optional,
- integer f2 optional
- }
- ...

- var r e1 := {};
- ...
- ------
- value of 'e1' is <unbound> instead of {f1 := <unbound>, f2 := <unbound>}

- **According to the TTCN-3 standard, a record/set value is unbound if all of its fields are unbound. This means that '<unbound>' and '{f1 := <unbound>, f2 := <unbound>}' are identical. In TITAN the unbound record/set value is always logged as '<unbound>'.**

BUG 540052 - GET LIST OF TSPS FROM TTCN BINARY

- Hi,
- In our TTCN test case execution Framework, we need to find the list of tsps in a TTCN binary. Now, we use 'strings' command and some grepping to find the list, and it takes a lot of time (close to 10 seconds), which makes our execution FW slow.

- To find list of test cases, we use "<ttcn binary> -l" and it takes a couple of seconds to get the result. Could such efficient functionality be implemented for tsps in Titan?

New command line option '-p' implemented for TTCN binaries (both in single mode and parallel mode), which lists all module parameters.

BUG 544632 - DUPLICATE IDENTIFIER USAGE CAUSES FATAL ERROR

- module test2 {
- template T T(template integer p := ?) := p
- const T c := { 1 };
- }
- causes the following error:
- AST_ttcn3.cc:8548: error: FormalPar::get_defval().
- **Fixed.**
- **Negative test added to function_test/Semantic_Analyser/TTCN3_SA_ttcn3adhoc_TD.script**.

BUG 544828 - CLASHING CODECS CAUSE CODE GENERATION ERROR WITH LEGACY CODEC HANDLING

- Reference to a non-existent element XER descriptor is generated if the record of/set of has XER encoding, but the element type doesn't. This can happen in some special cases with the legacy codec handling, when multiple codecs are declared for parts of a structured type.

- This can be reproduced with the following 3 modules:

```
A.ttcn
======
module A {

import from B all;

type record BigRec {
  RoRec f
}
with {
  variant (f) "JSON: name as f";
  encode "JSON";
}

type record BigRec2 {
  RoRec f
}

}
```

BUG 544828 - CLASHING CODECS CAUSE CODE GENERATION ERROR WITH LEGACY CODEC HANDLING

```
B.ttcn
======
module B {

import from C all;

type record Rec {
  Enum x
}
with {
  variant (x) "JSON: name as x";
  encode "JSON";
}

type record of Rec RoRec;
}
with {
  encode "XML";
  variant "namespace as 'www.somewhere.com'";
}


C.ttcn
======
module C {

type enumerated Enum { One(1), Two(2) }
with {
  variant "FIELDLENGTH(2)"
}

}
```

BUG 544828 - CLASHING CODECS CAUSE CODE GENERATION ERROR WITH LEGACY CODEC HANDLING

- A.ttcn must also appear before B.ttcn in the compiler's arguments. The location of C.ttcn is irrelevant.

- Fixed.
- The reference to the element type's XER descriptor is no longer generated if the element type doesn't have XER encoding.

BUG 544861 - PRINTING SYMBOLIC VERSION IN 'COMPILER -V'

- See https://www.eclipse.org/forums/index.php/t/1097649/

- Patch committed.
- The symbolic version string was also added to all other TITAN executables, including the ones built from the generated code and runtime library.

BUG 544855 - ISVALUE MIGHT THROW DTE

- According to the standard isvalue needs operate safely similar to isbound,ischoosen and ispresent.

- But right now in Titan, this is not the case.
- example:
- type component CT {}
- type union unionOper_myunion1 {integer x1, float x2};

- testcase akarmi () runs on CT {
- var unionOper_myunion1 x:={ x2:=1.0};
- if (isvalue(x.x1)) {setverdict(fail, __LINE__);}
- else {setverdict(pass, __LINE__);}
- }
- control {
- execute(akarmi());
- }
- Please note as in Titan it is possible to catch DTE.
- For this reason making isvalue operate safely can cause backward incompatibility, as it will no longer throw a DTE.
- However such situations would have been already problematic to be present in testing code.

- **Change implemented.**

BUG 544978 - ERROR WHEN FORCED SEOF GENERATION IS SET

- When using the -F flag (forced generation of Seof types) while compiling this code, the compiler runs into an error.

- The code:
- "
- type record of integer ro_integer_base;
- type ro_integer_base ro_integer;

- type record IMS_Configuration_TC_param
- {
- charstring name,
- charstring trafficCase
- }

- type record of integer Integers;
- type Integers EPTF_IntegerList;

- private function f_IMS_TC_Generic_getIpAndPortRanges(in IMS_Configuration_TC_param pl_IMS_LGen_configuration,
- out EPTF_IntegerList pl_lowIp, out EPTF_IntegerList pl_highIp, out integer pl_lowPort, out integer pl_highPort, out integer pl_pppLow, out integer pl_pppHigh) return integer {
- return 0;
- }

BUG 544978 - ERROR WHEN FORCED SEOF GENERATION IS SET

```
function something() {
  var IMS_Configuration_TC_param v_IMS_LGen_configuration;

  var ro_integer vl_lowIp, vl_highIp, vl_ip;
    var integer vl_basePort, vl_lowPort, vl_highPort;
    var integer pppLow, pppHigh;

    if (0 != f_IMS_TC_Generic_getIpAndPortRanges(v_IMS_LGen_configuration, vl_lowIp, vl_highIp, vl_lowPort, vl_highPort, pppLow, pppHigh)) {
      //f_IMS_Logging_error("Failed to get IP and port ranges for " & v_IMS_LGen_configuration.name);
    }
}
"
```

The error message:

"Value.cc:15329: error: Value::get_single_expr()"

BUG 544978 - ERROR WHEN FORCED SEOF GENERATION IS SET

- The fault doesn't have much to do with the compiler option '-F'. It does, however, require the option '-R' (runtime 2), since the fault is in the code generation for parameters that need type conversion. It can be reproduced with the following code:

```
type record Rec {
  integer x
}

type record of Rec RecList1;
type record of Rec RecList2;

function f(inout RecList1 x) return integer {
  return 0;
}

control {
  var RecList2 x;
  if (0 != f(x)) {
    log("1");
  }
}
```

- With the compiler option '-F' the fault also occurs for records of/sets of basic types (such as the 'record of integer' types from the previous example).
- Fixed.
- Regression test added under regression_test/compileonly/Bug544978.

BUG 545308 - LEGITIMIZE COMPILER OPTION '-0'

- Compiler option '-0' (used together with option '-s') is currently a secret option used by ETSI. It should be made public (i.e. it should be documented and added to the Eclipse plug-in options).
- Added option '-0' to the compiler usage printout, the compiler's manual page and the reference guide.
- Added a new check within the compiler that disallows the usage of option '-0' without option '-s'.
- Added two new entries in the TPD makefile settings that govern the compiler options '-s' and '-0' (<semanticCheckOnly> and <disableAttributeValidation>).

- Documented compiler option '-0' and the new TPD entries in the reference guide.

BUG 544608 - REPORT THE ASN.1 FEATURE "EXTENSIBILITY IMPLIED" AS NOT YET SUPPORTED

- Titan currently does not report the extensibility implied ASN.1 feature as not yet supported.
- It would be better if this warning would be displayed for the user, to forego misunderstandings later.
- Warning added.

BUG 541693 - CONSTRAINTS ON ASN.1 INTEGERS WITH NAMED VALUES IGNORED BY OER ENCODER

- In the following example, the OER encoder ignores the integer type's 2 constraints if they are specified separately, in 2 different types (the value is incorrectly encoded with a length indicator before its actual encoding). If the 2 constraints are specified in the same type, then the encoding is correct (no length indicator).

- ASN.1 file:
- ----------
- Test
- DEFINITIONS AUTOMATIC TAGS ::=
- BEGIN
- Int1 ::= INTEGER {v1(1)} (v1)
-
- IntBase ::= INTEGER {v1(1)}
- Int2 ::= IntBase (v1)
-
- x1 Int1 ::= v1
- x2 Int2 ::= v1
- END

BUG 541693 - CONSTRAINTS ON ASN.1 INTEGERS WITH NAMED VALUES IGNORED BY OER ENCODER

- TTCN-3 file:
- ----------
- module test {

- import from Test all;

- external function f_enc1(in Int1 x) return octetstring
- with { extension "prototype(convert) encode(OER)" }

- external function f_enc2(in Int2 x) return octetstring
- with { extension "prototype(convert) encode(OER)" }

- control {
- action(f_enc1(x1)); // '01'O
- action(f_enc2(x2)); // '0101'O
- }

- }
- Fixed OER-related semantic check to take the type constraints of all types into account, not just the last referenced type.

- Tests added to regression_test/OER.

BUG 545802 - LOG(OMIT) CAUSES COMPILER CRASH

- log(omit) or log2str(omit) causes the compiler to crash with a fatal error ('Value.cc:11426: error: log2str argument type').

- Example code:
- module test {

- control {
- var charstring x := log2str (omit);
- log(omit);
- }

- }
- Fixed.

BUG 546231 - RAW DEC: LENGTH PASSED INCORRECTLY TO RECORD FIELD OF RECORD-OF TYPE

```
module proba{

external function dec_R2(in octetstring stream) return R2
with { extension "prototype(convert)" extension "decode(RAW)" }

type record R2{
  integer   lengthfield,
  record of integer   listfield,
  integer   last_field
} with {
  variant (lengthfield) "LENGTHTO(listfield)"
  variant (lengthfield) "UNIT(elements)"
}

control{
  log(dec_R2('0000'O))    // causes segmentation fault, should be{ lengthfield := 0, listfield := { }, last_field := 0 }
  log(dec_R2('010102'O))  // decoding error, should be { lengthfield := 1, listfield := { 1 }, last_field := 2 }
}

} with {
  encode "RAW"
}
Fixed in titan.core.

Regression tests added.
```

Bug 546800 - Define function type which parameters with timer would get compiler error

- I defined a function type with a timer parameters, it would failed when I compiled the TTCN file contains it.

- // TTCN file 'test.ttcn'
- module test
- {

- type function test_func(timer T); // remove this line could pass compile

- function f_test(timer T)
- {
- log("just for test.");
- }

- }

- $> compiler test.ttcn
- ...
- Notify: Checking modules...
- Notify: Generating code...
- FATAL ERROR: /home/efuwwei/fuwei/titan/bin/compiler: In line 8382 of AST_ttcn3.cc: FormalPar::get_Type()
- make: *** [compile] Aborted
- Fixed. The compiler should no longer crash when it encounters a timer parameter in a function type.
- Tests added to regression_test/functionReference/FuncRef.ttcn.

BUG 546286 - DISPLAY WARNINGS FOR 'MAP PARAM' AND 'UNMAP PARAM'

- The TTCN-3 parser should display a warning for the 'map param' and 'unmap param' clauses from port type declarations. Currently the parser does not recognize them and displays parsing errors.
- The map/unmap parameters feature has been fully implemented.
- Tests have been added to regression_test/map_param and function_test/SemanticAnalyzer/map_param.
- Usage of the feature has been documented in the API guide.

BUG 547307 - BACKWARD INCOMPATIBILITY CAUSED BY OCTETSTRING ALIGNMENT CHANGE IN RAW CODEC

- The way the variant attribute 'ALIGN' works has been changed recently for octetstrings.
- See https://bugs.eclipse.org/bugs/show_bug.cgi?id=533767.

- This also changed the way octetstrings with default alignment (i.e. those with no 'ALIGN' variant attribute) are encoded/decoded, which has caused backward incompatibilities.

BUG 547318 - COLORIZE COMPILER ERROR/WARNING MESSAGES

- Suggestion and initial implementation by Harald Welte.
- See https://review.gerrithub.io/#/c/laf0rge/titan.core/+/454000/

BUGZILLA JANUARY

Bug ID	Product	Component	Assignee	Status	Resolution	Summary	Changed
538482	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Type descriptor generation problems caused by OER fix	12/12/2018 5:57
541748	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	xsd2ttcn: Name clashes and references not handled when using single module	12/14/2018 5:45
543155	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	New error behavior option for data remaining after decoding	1/4/2019 9:34
542610	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Translation ports across multiple Eclipse projects	1/14/2019 11:50
539612	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Ubuntu 18.04, gcc 7, c++11 compatibility	1/17/2019 5:59
517843	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Support for multiple encodings	1/21/2019 8:36

BUGZILLA FEBRUARY

Bug ID	Product	Component	Assignee	Status	Resolution	Summary	Changed
539514	Titan	Core	botond.baranyi@ericsson.com	CLOSED	FIXED	Real-time testing in TITAN	2/14/2019 11:28
542610	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Translation ports across multiple Eclipse projects	2/14/2019 11:28
543155	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	New error behavior option for data remaining after decoding	2/14/2019 11:28
537885	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Invalid initialization of reference (c++ error message); @fuzzy testcase parameter with default value.	2/14/2019 11:30
537979	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	@fuzzy as formal parameter: unspecified limitation of feature	2/14/2019 11:30
539612	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Ubuntu 18.04, gcc 7, c++11 compatibility	2/14/2019 11:30
537922	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Incorrect order in record generated by xsd2ttcn	2/14/2019 11:31
529443	Titan	Core	botond.baranyi@ericsson.com	CLOSED	FIXED	Extend 'default' JSON attribute for structured types	2/14/2019 11:33
539393	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	XER enc: encvalue uses incorrect type descriptor	2/15/2019 9:48
537328	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Bad assignment for records with optional fields	2/15/2019 10:26
540052	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Get list of TSPs from TTCN binary	2/20/2019 8:45
544632	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Duplicate identifier usage causes fatal error	2/21/2019 8:28
544828	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Clashing codecs cause code generation error with legacy codec handling	2/26/2019 11:41
544861	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Printing symbolic version in 'compiler -v'	2/27/2019 4:40

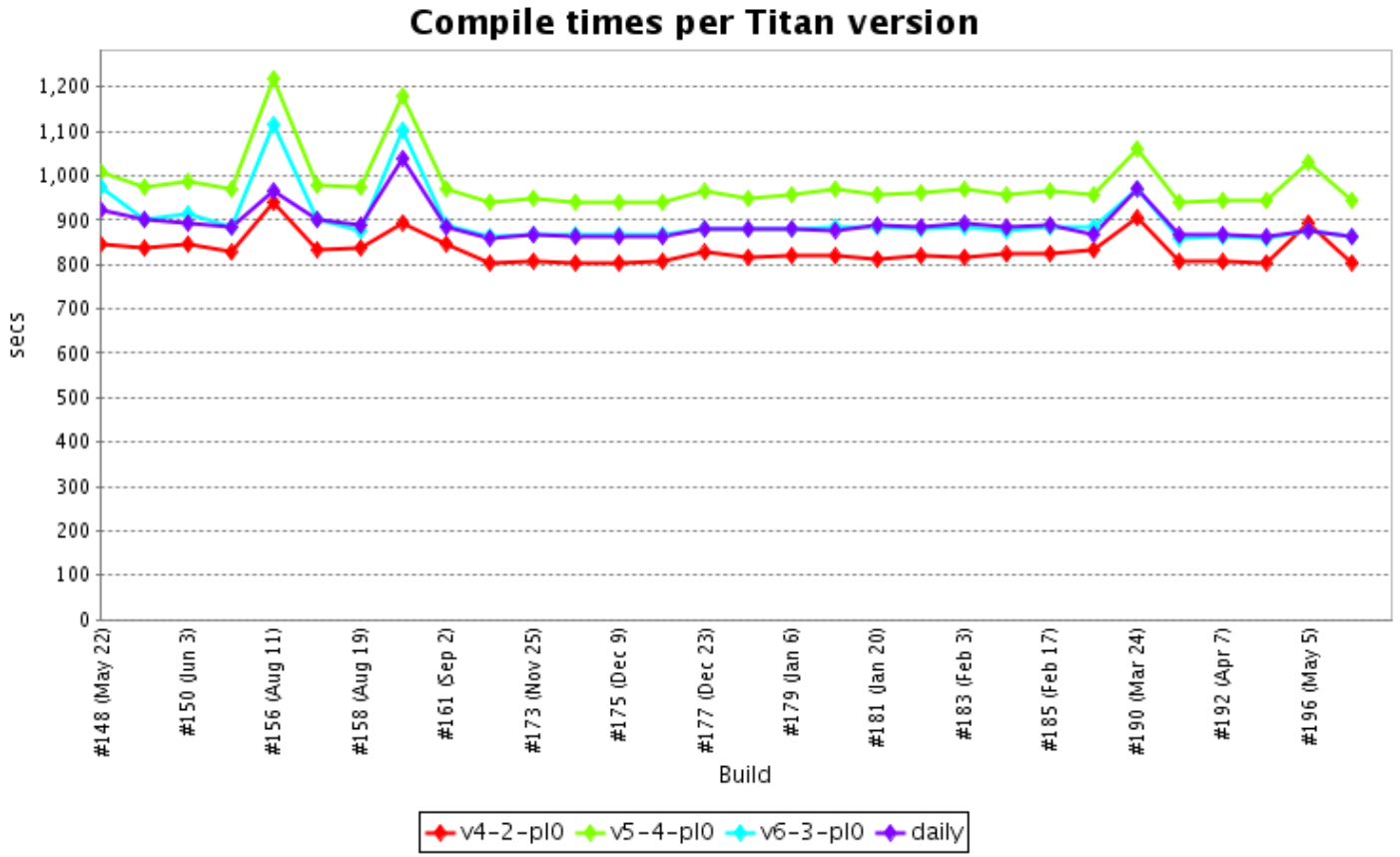
BUGZILLA MARCH

Bug ID	Product	Component	Assignee	Status	Resolution	Summary	Changed
544855	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	isvalue might throw DTE	3/1/2019 7:52
544978	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	error when forced seof generation is set	3/12/2019 11:51
545308	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Legitimize compiler option '-0'	3/13/2019 10:08
544608	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	report the ASN.1 feature "extensibility implied" as not yet supprted	3/18/2019 11:33
541693	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Constraints on ASN.1 integers with named values ignored by OER encoder	3/21/2019 5:30
545802	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	log(omit) causes compiler crash	3/29/2019 8:22

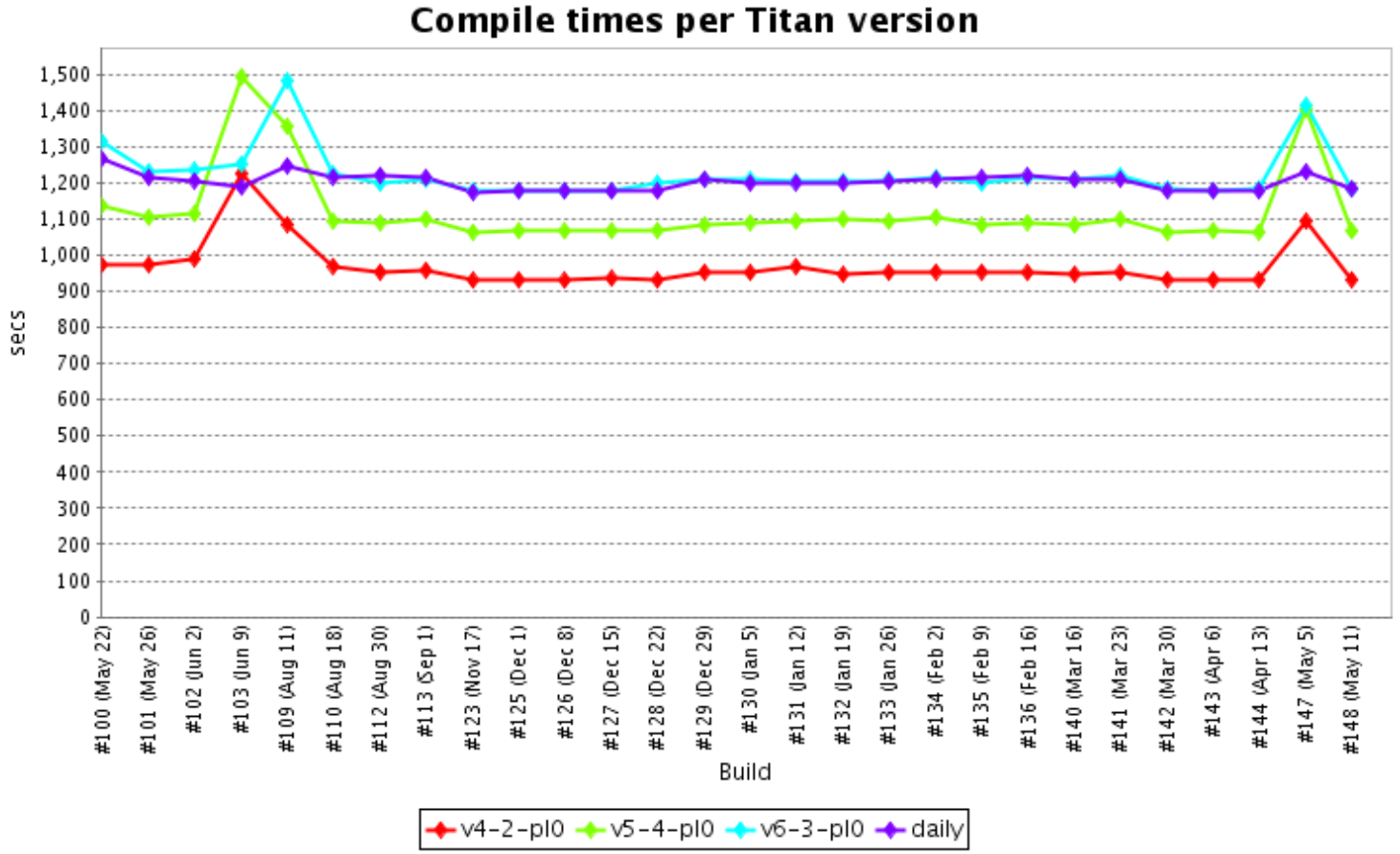
BUGZILLA APRIL&MAY

546231	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	RAW dec: length passed incorrectly to record field of record-of type	4/9/2019 11:30
546800	Titan	Core	titan-inbox@eclipse.org	RESOLVED	FIXED	Define function type which parameters with timer would get compiler error	4/29/2019 6:57
546286	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Display warnings for 'map param' and 'unmap param'	5/7/2019 8:15
547307	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Backward incompatibility caused by octetstring alignment change in RAW codec	5/15/2019 7:19
547318	Titan	Core	titan-inbox@eclipse.org	NEW	---	Colorize compiler error/warning messages	5/15/2019 7:57

COMPILE TIMES RT1



COMPILE TIMES RT2



ERICSSON