TITAN 7.1.pl0 Change log

Summary

In this release we upgraded our license to Eclipse Plugin License 2.0, introduced both major new features and backward incompatibilities.

We upgraded our license to Eclipse Plugin License 2.0

- 536130 License upgrade to EPL 2.0

New Features on the C side

On the C side the most important new feature is the support of the Object-Oriented Extension of the TTCN-3 standard
(please note, it is a feature in progress with some features still missing as described in the Bugzilla bug):

- 552011 Object-oriented features

New Features on the Java side

On the Java side we have greatly improved the processing speed by parallelizing some of the operations:

- 559327 Titanium code smell checkers could be run in parallel
- 559544 parallelizing the semantic checking
- 559919 improve parallelism of syntactic checking (parsing) in case of project hierarchies

Also on the Java side we have added support for JSON encoding/decoding:

- 559438 ttcn2java JSON encoder/decoder for TTCN-3 basic types
- 559439 ttcn2java JSON encoder/decoder for enum
- 559440 ttcn2java JSON encoder/decoder for union
- 559441 new feature: ttcn2java JSON encoder/decoder for recordof
- 559442 new feature: ttcn2java JSON encoder/decoder for record
- 559443 ttcn2java JSON encoder/decoder built-in functions

Backward Incompatibilities

- 559422 'omit' matching mechanism shouldn't be allowed for union fields
- 561653 Compilation issue in the Java code generator when used with very large modules

Bugfixes

- 559035 titan.TestPorts.IPL4asp R.30.E doesn't build (ERROR_LENGTH)
- 559350 the "Check code for code smells" menu item does not appear
- 559408 ttcn2java: version extension handling error, java code is not generated
- 559580 ttcn2java: Wrong generated code: JSON cannot be resolved to a variable
- 559630 bugfix: TitanOctetString.JSON_encode() returns wrong encoded value
- 559789 ttcn2java Wrong generated code if in ASN1 hexadec value assigned to var with type of ANY
- 559978 potential coding issues to be checked with unnamed temporal objects
- 559652 False semantic error for encvalue() of value of anytype
- 559371 Faulty code for select union
- 560347 OER: encoding unbound optional field causes error in RT2
- 560464 Memory leak when JSON decoding binary strings
- 553584 Support for ETSI's JSON module
- 560937 Bitstrings don't clear unused bits after JSON decoding
- 559413 ttcn2java: Java code generated from empty module is erroneous
- 560851 can't build asn1 which are generated by asn1 compile

Bugfixes

- 560973 Errors overlooked checking templates not sufficiently defined
- 562488 RAW codec: Incorrect decoding of an unaligned bit-field (BIT5)
- 561653 Compilation issue in the Java code generator when used with very large modules
- 560016 ttcn2java: compilation error if ASN.1 enum def contains reference
- 559944 titan.ProtocolModules.MobileL3_v13.4.0 git repository disappeared
- 559943 titan.ProtocolModules.LLC_v7.1.0 git repository disappeared
- 546045 MQTT Decoding of remaining length
- 559942 titan.TestPorts.GPIO repository disappeared
- 563218 opening the call hierarchy might mark some references as erroneous
- 559373 host name lookup failure during make install
- 520933 Add info about translation ports in the online index of TITAN
- 514844 Update help page with ports with translation capability
- 529895 Designer: support syntax for multiple encodings
- 560188 enhance helper function calls in RAW encode decode functions
- 553271 call hierarchy in the designer

New Feature Details

BUG 536130 - License upgrade to EPL 2.0

The Eclipse Community change the license version to EPL 2.0

Implemented

Now all titan related product have EPL 2.0 license.

BUG 552011 - Object-oriented features

Implement object-oriented features according to chapter 5 of the standard extension:
https://www.etsi.org/deliver/etsi_es/203700_203799/203790/01.01.01_60/es_203790v010101p.pdf

Implemented with limitations.

Object-oriented features require the compiler option '-k', as they introduce new keywords.

The following features limitations:

- Classes cannot be embedded into other structures (e.g. records), only into other classes.
- Exceptions are not yet supported.
- Class members of ports and port array types are not supported.
- The 'select-class' statement, 'of' operator and casting operator are not yet supported.

BUG 559327 - Titanium code smell checkers could be run in parallel

As the code smell checkers of Titanium are stateless, they could be run in parallel.

This would allow faster code quality checking, of large projects, on multi core processors.

Implemented.

On a 4 core 8 thread laptop, we see a 2-4* improvement in processing speed.

BUG 559544 - parallelizing the semantic checking

It might be hard, but also beneficial to parallelize the semantic checking part of the Designer's operations.

In multi core/thread environments (present on almost all laptops and servers nowadays) this could further speed up the checking of the code.

Implemented.

On synthetic tests reached ~ 3.125* speed compared to single threaded checking.

In real life code bases we observed around 1.2* - 2* speed improvement.

In the synthetic test project the speed was valid for the whole analysis.

Starting from a closed project, reaching checked state was around 3* faster, than doing the same in the command line.

BUG 559919 - improve parallelism of syntactic checking (parsing) in case of project hierarchies

Currently the Designer is able to parse in parallel the source files contained within projects, to improve processing speed.

However recent trends indicate, that large project hierarchies (containing several projects) becoming more and more widespread to address the complexity of ever growing source code base.

Implemented.

In the first implementation on a 4 core laptop we have seen 20% - 30% reduction in the overall time required to check the project hierarchy.

BUG 559438 - ttcn2java JSON encoder/decoder for TTCN-3 basic types

ttcn2java JSON encoder/decoder for TTCN-3 basic types.

Implemented.

BUG 559439 - ttcn2java JSON encoder/decoder for enum

Implemented.

BUG 559440 - ttcn2java JSON encoder/decoder for union

Implemented.

BUG 559441 - new feature: ttcn2java JSON encoder/decoder for recordof

Implemented.

BUG 559442 - new feature: ttcn2java JSON encoder/decoder for record

BUG 559443 - ttcn2java JSON encoder/decoder built-in functions

Implemented.

Details of backward incompatible changes

BUG 559422 - 'omit' matching mechanism shouldn't be allowed for union fields

Example:

```
module UnionAndOmitBug {
type component CT{}
type union UU {
 integer a,
 charstring s
}
testcase tc_omitInUnion() runs on CT {
 var template UU vtu := { a := 42 }
 vtu.s := omit; //DTE in java, accepted in cpp //suggested: semantic error: 'omit' value is not allowed in this context + runtime: better output
}
testcase tc_semanticErrorExpected2() runs on CT {
 var template UU vtu := { a := omit } //ok: error marker: 'omit' value is not allowed in this context; compiler error in cpp, runtime error in java.
} }
```

Fixed.

Template variable assignments have been restricted to no longer allow 'omit' for union fields an non-optional fields of records/sets..

BUG 561653 - Compilation issue in the Java code generator when used with very large modules

If a TTCN-3 module contains thousands of types each of which has encodings, the generated code has too many typedescriptors in the class belonging to the module to compile.

So far this only happened in an extreme case, where with the DIAMETER generator it was possible to create a module with ~1600 type definitions.

Fixed.

For type for which we generate a class to represent them, the type descriptor will be generated into the body of this class

Bug Details

BUG 559035 - titan.TestPorts.IPL4asp R.30.E doesn't build (ERROR_LENGTH)

In commit 7206152ca2572f0bb334c82d77f5436e76b1f6bf, the following line of code was added:

> const PortError IPL4_ERROR_LENGTH := ERROR_LENGTH;

I don't know where that ERROR_LENGTH definition should be coming frrom. As soon as I use E.30.E, compileation will fail due to this undefined ERROR_LENGTH.

Fixed.

the test port has a dependency from SocketAPI; the latest of SocketAPI has not been pushed to its' repo.

It should be fixed by now.

BUG 559350 - the "Check code for code smells" menu item does not appear

The "Check code for code smells" menu item does not appear for some reason.

This way if automatic quality checking is not turned on, users can not manually ask for it.

FIXED.

BUG 559408 - ttcn2java: version extension handling error, java code is not generated

INVALID

1. Originally the ttcn2java compilation of this ttcn module gave a NPE:

java.lang.NullPointerException

at org.eclipse.titan.designer.AST.TTCN3.attributes.ModuleVersionAttribute.parse(ModuleVersionAttribute.java:55)

at org.eclipse.titan.designer.AST.TTCN3.definitions.TTCN3Module.analyzeExtensionAttributes(TTCN3Module.java:602)

at org.eclipse.titan.designer.AST.TTCN3.definitions.TTCN3Module.check(TTCN3Module.java:479)

at org.eclipse.titan.designer.AST.brokenpartsanalyzers.BrokenPartsChecker.generalChecker(BrokenPartsChecker.java:79)

The reason was that in the line 55, versionToken.getTtcnName();

the versionToken was null.

2. After modification the compiler does not give error but silently does not generate code. The expected behavior is at least a compilation error message, as in cpp does: "error: at or before token `>': syntax error, unexpected '>', expecting Identifier". Or similar

BUG 559580 - ttcn2java: Wrong generated code: JSON cannot be resolved to a variable

Created attachment 281606 [details]

First example file

The generated java code of the attached ttcn and asn1 code is erroneous.

FIXED

Import was missing:

aData.addBuiltinTypeImport("JSON");

BUG 559630 - bugfix: TitanOctetString.JSON_encode() returns wrong encoded value

- Fixed.

BUG 559789 - ttcn2java Wrong generated code if in ASN1 hexadec value assigned to var with type of ANY

In an ASN.1 file, if the hexadecimal value is assigned to a value of type ANY, then the ttcn2java code generator created an erroneous code or a semantically correct code which will cause runtime error

For example:

```
x ANY ::= 'ABBA'H  --erroneous java code
or
Object1 ::= SEQUENCE {
  id OBJECT IDENTIFIER,
  data ANY
}
c-obj Object1 ::= {
  id { joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0) },
  data 'DEADBEEF'H  --runtime error
}
```

FIXED.

BUG 559978 - potential coding issues to be checked with unnamed temporal objects

In some locations the error context tracking objact is contructed without a local variable.

Which instructs the C++ compiler to destruct it at the end of the full expression, instead of the scope it is located in.

for example Statement.cc, "chk_signature_ref" has the following code:

"Error_Context(p_ref, "In signature");"

This is not the intended behaviour, and also is an incorrect use of our related code pattern.

FIXED

BUG 559652 - False semantic error for encvalue() of value of anytype

```
module AnytypeMarkerTest {
type integer I;
type charstring CS;
control {
 var anytype a;
 a.I := 1;
 var bitstring bs := encvalue(a);//marker on this line
}
} with {
encode "JSON";
extension "anytype I, CS";
}
```

Marker text:

```
============
No coding rule specified for type `@AnytypeMarkerTest.anytype'   AnytypeMarkerTest.ttcn   /TR_2020_01_ttcn2java/src   line 9   on-the-fly semantic markers
```

FIXED.

BUG 559371 - Faulty code for select union

Faulty code generated for select union in the case of optional field

The generated code:

```
switch(const_cast< const HTTP2__headers__struct&>(vl__headers).content__type().get_selection()) {
```

should be:

```
switch(const_cast< const HTTP2__headers__struct&>(vl__headers).content__type()().get_selection()) {
```

Fixed.

New test case added to regression_test/selectUnion.

BUG 560347 - OER: encoding unbound optional field causes error in RT2

See regression_test/EncodeDecode/OER_EncDec.

testcase tc_OER_SequenceOptionalExtended() runs on Test_CT {

This and 3 other test cases end with the error 'Encoding an unbound optional value.' in Runtime 2. They pass in Runtime 1.

Fixed.

Both runtimes now successfully encode the mentioned records.

BUG 560464 - Memory leak when JSON decoding binary strings

See regression_test/EncodeDecode/JSON/JsonBasicTest/JsonBasicEncDecTest_bs.ttcn.

In several of these testcases the previous value of the bitstring/hexstring/octetstring is not freed before the JSON decoding result is assigned. This only occurs when decoding a bound value with 'decvalue' or a decoder function of prototype FAST, SLIDING or BACKTRACK. (Prototype CONVERT performs the decoding on an empty value and copies the result to the initial variable.)

The same might be true for large integers (>2^31), but that cannot be detected with a debug mode TITAN, since the memory allocation of BIGNUMs is handled by an external library.

Fixed.

Additional test case added for large integers.

BUG 553584 - Support for ETSI's JSON module

Support should be added for the JSON module in Annex A in Part 11 of the TTCN specification (ETSI ES 201 873-11 V4.8.1 (2018-05)).

Currently many of the variant attributes are not recognized by our JSON codec.

The module also contains the string "\", which is not supported by TITAN (in TITAN, the backslash character is escaped: "\\").

Fixed a bug, where the new keywords were not accepted as field aliases by the attribute 'name as ...'.

(e.g. 'name as string' caused an error, because 'string' is now a keyword)

BUG 560937 - Bitstrings don't clear unused bits after JSON decoding

According to Gabor Szalai, the bitstrings in the PDU do not clear their ususued bits after the JSON decoding. Logging them produces correct results, but encoding them with RAW causes incorrect results. The operator '==' seemingly clears unused bits, and RAW encoding is successful afterwards.

The same may be true for hexstrings.

Fixed.

(Could not reproduce the issue with hexstrings, but the same fix was added to the hexstring's decoder, too.)

Regression test case added to regression_test/multipleEncodings.

BUG 559413 - ttcn2java: Java code generated from empty module is erroneous

The java code generated from the attached module is erroneous as java file:

Description Resource Path Location Type

TTCN_EncDec cannot be resolved to a variable EmptyMod.java /TR_2020_01_ttcn2java/java_src/org/eclipse/titan/TR_2020_01_ttcn2java/generated line 349 Java Problem

FIXED

BUG 560851 - can't build asn1 which are generated by asn1 compile

BUG 560973 - Errors overlooked checking templates not sufficiently defined

We found out a possible issue doing syntax checks with Titan on our TTCN3 deliveries at ETSI.

The point is that in some situations, templates defined from asn1 types are not fully checked and only with other tools we are able to find these errors.

As we got feedback that version 6.5.pl1 should also report these warnings, please find below the command line arguments we use to call the tool:

-s0 -d -R

BUG 562488 - RAW codec: Incorrect decoding of an unaligned bit-field (BIT5)

As was requested on Eclipse Community Forums, where I initially described the problem, I am opening a bug report here. To get more context information, please see: https://www.eclipse.org/forums/index.php/m/1826490/#msg_1826490.

Explained

Applying BYTEORDER(last) to the whole module seems to work.

```
module TestModule {
// ...
} with { encode "RAW" ; variant "FIELDORDER(msb)" variant "BYTEORDER(last)" }
```

I intuitively did this in the attached example just before uploading it here, but forgot to test again afterwards. The problem is solved now.

BUG 561653 - Compilation issue in the Java code generator when used with very large modules

We have found a limitation in the way we currently generate the code in the Java code generator.

If a TTCN-3 module contains thousands of types each of which has encodings, the generated code has too many typedescriptors in the class belonging to the module to compile.

So far this only happened in an extreme case, where with the DIAMETER generator it was possible to create a module with ~1600 type definitions.

Fixed.

For type for which we generate a class to represent them, the type descriptor will be generated into the body of this class.

BUG 560016 - ttcn2java: compilation error if ASN.1 enum def contains reference

The java code generation stops with exception.

ASN1_Enumerated_Type.java:834 tries to cast a reference to

fields.add(new Enum_field(tempItem.getId().getName(), tempItem.getId().getDisplayName(), ((Integer_Value)tempItem.getValue()).getValue()));

FIXED

BUG 559944 - titan.ProtocolModules.MobileL3_v13.4.0 git repository disappeared

Osmocom.org automatic test suites are failing as the MobileL3 protocol module has disappeared. It cannot be reached either over https nor over git protocol

https://git.eclipse.org/r/titan/titan.ProtocolModules.MobileL3_v13.4.0.git simply renders "Not Found”

*** This bug has been marked as a duplicate of bug 559943 ***

BUG 559943 - titan.ProtocolModules.LLC_v7.1.0 git repository disappeared

Osmocom.org automatic test suites are failing as the LLC protocol module has disappeared. It cannot be reached either over https nor over git protocol

https://git.eclipse.org/r/titan/titan.ProtocolModules.LLC_v7.1.0.git simply renders "Not Found"

I think the initial issue was caused by the backend file store that fell over a few days ago. But I"ve taken a look in the logs and it's not reporting anything regarding this repo/URL .

Frankly I'm a little surprised that it worked at all since I wouldn't expect to be able to 'browse' a repo via at that URL, since it's a 'clone' URL.

-M.

BUG 546045 - MQTT Decoding of remaining length

While using the MQTT ProtocolModule for sending and receiving large payloads I discovered a bug in the decoding of the remaining length field. The decoding works perfectly fine for remaining lengths encoded as 1, 2 and 3 bytes, but fails for 4 bytes.

I was able to pinpoint the reason in negative_testing/MQTT_v3_1_1_EncDec.cc

However, this bug seems to originate from the MQTT v3.1.1 specification itself and is already fixed in the MQTT v5 specification.

the short story is that TCP does not have the concept of message length

of the protocol which uses TCP as a transport;

so the application protocol has to tell the TCP layer (in our case the IPL4 test port) how the messages are delimited. This is done by registering a callback function which helps the IPL4 to calculate message boundaries.

Each protocol module intending to use TCP as transport will have to have this function added. For MQTT it's f_calc_MQTT_length in MQTT_v3_1_1_IPL4SizeFunction.ttcn.

If the messages are short and rare, there's a good chance that they are not segmented, so they will arrive complete. But if they will become longer , segmentation kicks in and in lack of the callback function TCP will send the segments to the upper layer. With the callback registered, the IPL4 will re-establish message boundaries.

I'm afraid I missed this in the MQTT example I have posted. But in other places , HTTP/2, Websocket I have included it.

I hope this helps

BUG 559942 - titan.TestPorts.GPIO repository disappeared

Our automatic test suites at osmocom.org are currently failing as they can no longer clone the titan.TestPorts.GPIO repository from git://git.eclipse.org/gitroot/titan/titan.TestPorts.GPIO.git where it used to be available for years.

Is the removal intentional?

This also looks like related to Bug 559943.

The links from our download site work correctly.

https://git.eclipse.org/r/plugins/gitiles/titan/titan.TestPorts.GPIO

My best guess is that eclipse.org maintainers might have rearranged something that resulted in the change in the git clone address.

*** This bug has been marked as a duplicate of bug 559943 ***

BUG 563218 - opening the call hierarchy might mark some references as erroneous

When opening the call hierarchy from the TTCN-3 editor some references in the project (including in other modules) are becoming marked as semantically erroneous.

Stating that they are references to parameterized definitions without actual parameter list, which in the actual context is not actually needed or allowed.

FIXED

For the call hierarchy to work it has to traverse the semantic tree, to look for locations that might call the function it needs to find the callers for.

During this traversal while checking where each reference points to, it accidentally asked a check for the formal parameters of the references, unnecessarily.

BUG 559373 - host name lookup failure during make install

During make install titan might fail to run the hello world test on some systems.

FIXED

The script run during make install will always try to use the localhost as the target hostcontrollers should connect to (as both the hostcontroller and the Main controller are running on the same machine this should be safe)

BUG 520933 - Add info about translation ports in the online index of TITAN

Here:

http://ttcn.ericsson.se/download/help/titan_index.html

Help updated.

BUG 514844 - Update help page with ports with translation capability

- Help updated

BUG 529895 - Designer: support syntax for multiple encodings

Created attachment 272283 [details]

on-the-fly analyser false errors

Currently Titan Designer doesn't support the syntaxes to bind variant attributes to one or more encodings. The standard syntaxes are:

a) "<encode>"."<enc.instruction>" //binding the variant attribute to one encoding

b) {"<encode1","<encode2"}."<enc.instruction>" //binding the variant attribute to multiple encodings

The bigger problem is that when the Designer meets this syntax, it generates a huge number of false error messages. On the attached screenshot Designer's real problem is the dot in "XML"."name as uncapitalized" in line 40, but it gives false "There is no visible definition with the name '...' in module '..."' error messages for all fields of local types of the record (the types are present in the module and the code compiles with the compiler).

Titan compiler currently supports syntax a) only, but it is proposed the Designer supports both syntaxes to prevent future misalignment.

The syntax is now supported.

BUG 560188 - enhance helper function calls in RAW encode decode functions

To be able to RAW encode/decode very large types, some encoding/decoding helper functions are used. (otherwise the size of the generated code could exceed in the case of large types the limits of Java set for single functions).

While this works, the current implementation can be improved upon.

Current style:

"

```
if (sel_field <= 200) {
decoded_length = RAW_decode_helper_0_199(buff, limit, top_bit_ord, no_err, sel_field, first_call, force_omit);
} else if (sel_field <= 400) {
decoded_length = RAW_decode_helper_200_399(buff, limit, top_bit_ord, no_err, sel_field, first_call, force_omit);
} else if (sel_field <= 600)
...
```

"

This could be speed up by using a switch on the number gained by dividing the field number by the partitioning size.

fixed using the below mentioned style. If there are more than 200 fields in a union we will generate such helper functions that each handle 200 fields.

BUG 553271 - call hierarchy in the designer

- Only needed to be closed.

Bugzilla December - January

ID	Product	Comp	Assignee	Status	Resolution	Summary	Changed
559035	Titan	TestPort	titan-inbox@eclipse.org	RESOLVED	FIXED	titan.TestPorts.IPL4asp R.30.E doesn't build (ERROR_LENGTH)	2020-01-14
536130	Titan	Core	jeno.balasko@ericsson.com	CLOSED	FIXED	License upgrade to EPL 2.0	2020-01-17
559350	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	the "Check code for code smells" menu item does not appear	2020-01-21
559422	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	'omit' matching mechanism shouldn't be allowed for union fields	2020-01-22
559438	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	ttcn2java JSON encoder/decoder for TTCN-3 basic types	2020-01-23
559440	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	ttcn2java JSON encoder/decoder for union	2020-01-23
559443	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	ttcn2java JSON encoder/decoder built-in functions	2020-01-23
559408	Titan	Plug-ins	jeno.balasko@ericsson.com	CLOSED	INVALID	ttcn2java: version extension handling error, java code is not generated	2020-01-23
559580	Titan	Plug-ins	alovassy@gmail.com	CLOSED	FIXED	ttcn2java: Wrong generated code: JSON cannot be resolved to a variable	2020-01-27
559441	Titan	Plug-ins	arpad.lovassy@semcon.com	CLOSED	FIXED	new feature: ttcn2java JSON encoder/decoder for recordof	2020-01-28
559442	Titan	Plug-ins	arpad.lovassy@semcon.com	CLOSED	FIXED	new feature: ttcn2java JSON encoder/decoder for record	2020-01-28
559630	Titan	Plug-ins	arpad.lovassy@semcon.com	CLOSED	FIXED	bugfix: TitanOctetString.JSON_encode() returns wrong encoded value	2020-01-28

Bugzilla February

ID	Product	Comp	Assignee	Status	Resolution	Summary	Changed
559439	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	ttcn2java JSON encoder/decoder for enum	2020-02-04
559789	Titan	Plug-ins	jeno.balasko@ericsson.com	CLOSED	FIXED	ttcn2java Wrong generated code if in ASN1 hexadec value assigned to var with type of ANY	2020-02-06
559978	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	potential coding issues to be checked with unnamed temporal objects	2020-02-11
559652	Titan	Plug-ins	arpad.lovassy@semcon.com	CLOSED	FIXED	False semantic error for encvalue() of value of anytype	2020-02-11
559371	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Faulty code for select union	2020-02-21
560347	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	OER: encoding unbound optional field causes error in RT2	2020-02-24
560464	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Memory leak when JSON decoding binary strings	2020-02-25

Bugzilla March

ID	Product	Comp	Assignee	Status	Resolution	Summary	Changed
553584	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Support for ETSI's JSON module	2020-03-10
560937	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	Bitstrings don't clear unused bits after JSON decoding	2020-03-10
559413	Titan	Plug-ins	kristof.szabados@ericsson.com	CLOSED	FIXED	ttcn2java: Java code generated from empty module is erroneous	2020-03-12
560851	Titan	Core	botond.baranyi@ericsson.com	CLOSED	FIXED	can't build asn1 which are generated by asn1 compile	2020-03-31

Bugzilla April

ID	Product	Comp	Assignee	Status	Resolution	Summary	Changed
560973	Titan	Other	botond.baranyi@ericsson.com	RESOLVED	FIXED	Errors overlooked checking templates not sufficiently defined	2020-04-20
562488	Titan	Core	titan-inbox@eclipse.org	CLOSED	WORKSFORME	RAW codec: Incorrect decoding of an unaligned bit-field (BIT5)	2020-04-27

Bugzilla May

ID	Product	Comp	Assignee	Status	Resolution	Summary	Changed
561653	Titan	Plug-ins	titan-inbox@eclipse.org	RESOLVED	FIXED	Compilation issue in the Java code generator when used with very large modules	2020-05-14
560016	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	ttcn2java: compilation error if ASN.1 enum def contains reference	2020-05-14
559944	Titan	Protocol	titan-inbox@eclipse.org	CLOSED	DUPLICATE	titan.ProtocolModules.MobileL3_v13.4.0 git repository disappeared	2020-05-14
559943	Titan	Protocol	titan-inbox@eclipse.org	CLOSED	WORKSFORME	titan.ProtocolModules.LLC_v7.1.0 git repository disappeared	2020-05-14
546045	Titan	Protocol	titan-inbox@eclipse.org	RESOLVED	FIXED	MQTT Decoding of remaining length	2020-05-14
559942	Titan	TestPort	titan-inbox@eclipse.org	CLOSED	DUPLICATE	titan.TestPorts.GPIO repository disappeared	2020-05-14
563218	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	opening the call hierarchy might mark some references as erroneous	2020-05-15
559373	Titan	Core	titan-inbox@eclipse.org	CLOSED	FIXED	host name lookup failure during make install	2020-05-26
520933	Titan	Core	lenard.nagy@ericsson.com	CLOSED	FIXED	Add info about translation ports in the online index of TITAN	2020-05-26
514844	Titan	Other	lenard.nagy@ericsson.com	CLOSED	FIXED	Update help page with ports with translation capability	2020-05-26
529895	Titan	Other	titan-inbox@eclipse.org	CLOSED	FIXED	Designer: support syntax for multiple encodings	2020-05-26
559327	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	Titanium code smell checkers could be run in parallel	2020-05-26
559544	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	parallelizing the semantic checking	2020-05-26
559919	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	improve parallelism of syntactic checking (parsing) in case of project hierarchies	2020-05-26
560188	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	enhance helper function calls in RAW encode decode functions	2020-05-26
553271	Titan	Plug-ins	titan-inbox@eclipse.org	CLOSED	FIXED	call hierarchy in the designer	2020-05-26

Documentation

CRL 113 200/7 R1A		
109 21	CRL 113 200/7-1Uen Rev.A	PRI
109 47	CRL 113 200/7 Uen Rev.A	Release Notes
1/1531	CRL 113 200/7 Uen Rev.A	Install. Guide for....Test Executor
3/1531	CRL 113 200/7 Uen Rev.A	Install. Guide for....TITAN Designer and..... Eclipse IDE
1551	CRL 113 200/7 Uen Rev.A	Titanium Description
1/174 02	CRL 113 200/7 Uen Rev.A	Statement of Compliance for Eclipse Titan
2/174 02	CRL 113 200/7 Uen Rev.A	Statement of Compliance for use of XML schema in Eclipse Titan
1/198 17	CRL 113 200/7 Uen Rev.A	UG for Titan TTCN-3 Test Executor
2/198 17	CRL 113 200/7 Uen Rev.A	Programmers Tech. Reference Guide for the TITAN executor
4/198 17	CRL 113 200/7 Uen Rev.A	UG for the TITAN Designer for the Eclipse IDE
5/198 17	CRL 113 200/7 Uen Rev.A	UG of the TITAN Executor for the Eclipse IDE plug-in
6/198 17	CRL 113 200/7 Uen Rev.A	API Technical Reference for the TITAN executor
8/198 17	CRL 113 200/7 Uen Rev.A	Titan Executor API User Guide
9/198 17	CRL 113 200/7 Uen Rev.A	Programmers Tech. Reference Guide for Titanium
1/1551	CRL 113 200/7 Uen Rev.A	Titanium Refactoring Description
10/198 17	CRL 113 200/7 Uen Rev.A	Programmers' Technical ReferenceGuide for the Java side of the TITANTTCN-3 Toolset