

1

Tool Development and Support

SPARK Toolset Release Note—Release 7.5

EXM/RN
Issue: 1.1
Status: Definitive
29 May 2007

Originator

SPARK Team

Approver

SPARK Team Line Manager

Copyright

The contents of this manual are the subject of copyright and all rights in it are reserved. The manual may not be copied, in whole or in part, without the written consent of Praxis High Integrity Systems Limited.

The software tools referred to in this manual are the subject of copyright and all rights in them are reserved. The rights in these tools are owned by Praxis High Integrity Systems Limited, and it may not be copied, in whole or in part, without the written consent of this company, except for reasonable back-up purposes. The same proprietary and copyright notices must be affixed to any permitted copies as were affixed to the original. This exception does not allow copies to be made for others, whether or not sold, and none of the material purchased may be sold, given or loaned to another person or organisation. Under law copying includes translating into another language or format.

©1991-2007 Praxis High Integrity Systems Limited, 20 Manvers St, Bath BA1 1PX

Limited Warranty

Praxis High Integrity Systems Limited save as required by law makes no warranty or representation, either express or implied, with respect to this software, its quality, performance, merchantability or fitness for a purpose. As a result, the licence to use this software is sold 'as is' and you, the purchaser, are assuming the entire risk as to its quality and performance.

Praxis High Integrity Systems Limited accepts no liability for direct, indirect, special or consequential damages nor any other legal liability whatsoever and howsoever arising resulting from any defect in the software or its documentation, even if advised of the possibility of such damages. In particular Praxis High Integrity Systems Limited accepts no liability for any programs or data stored or processed using Praxis High Integrity Systems Limited products, including the costs of recovering such programs or data.

SPADE is a trademark of Praxis High Integrity Systems Limited

Note: The SPARK programming language is not sponsored by or affiliated with SPARC International Inc. and is not based on SPARC™ architecture.

1

Contents

1	Introduction	5
2	Contact Information	6
3	SPARK language changes	7
4	SEPR 2124 – Important Information for Users	8
4.1	Description	8
4.2	SPARK Examiner versions affected	8
4.3	Test case	8
4.4	Workaround	10
4.5	Recommended action	10
5	SEPR 1979 – Important Information for Users	12
5.1	Description	12
5.2	SPARK Examiner versions affected	12
5.3	Test case	12
5.4	Workaround	12
5.5	Recommended action	13
6	Examiner changes	14
6.1	Increase in per-file justification limit (SEPR 2112)	14
6.2	New RTE checks for Ada.Real_Time (SEPR 2107)	14
6.3	Hypotheses for components unchanged by a call (SEPR 2108)	14
6.4	Proof functions and abstract own variables (SEPR 2109)	14
6.5	Failure to initialize a protected object is a flow error (SEPR 2113)	14
6.6	Visibility of embedded package own variables from a for loop (SEPRs 1696, 2067)	14
6.7	Space allowed in Windows command line (SEPR 2028)	15
6.8	Accept annotation following plain loop (SEPR 2075)	15
6.9	Size clause value must be static (SEPR 2093)	15
6.10	Number of arguments for pragmas (SEPR 2096)	15
6.11	Examiner memory usage (SEPR 2123)	15
6.12	New warning control keywords: default_loop_assertions and real_rtcs (SEPR 2126)	15
7	SPADE Simplifier	16
7.1	Improved reasoning for inequalities involving transitivity (SEPR 2120)	16
7.2	Enhanced reporting of user defined rules (SEPR 2106)	16
7.3	SPARKSimp	16
8	SPADE Proof Checker	17
9	POGS	18
9.1	Description of user-defined proof rules and summary-only option (SEPR 2106)	18
10	SPARKFormat	19

1

10.1	Empty line in global annotation should not signal end of annotation (SEPR 2125)	19
11	SPARKMake	20
12	Backward incompatibilities	21
13	User manual change summary	22
13.1	Examiner User Manual	22
13.2	Simplifier User Manual	22
13.3	POGS User Manual	22
14	Limitations and known errors	23
14.1	Tool limitations	23
14.2	Known error summary	25
15	Change Summary from Release 2.0	26
15.1	Release 2.0 - November 1995	26
15.2	Release 2.1 - July 1996	26
15.3	Release 2.5 - March 1997	26
15.4	Release 3.0 - September 1997	26
15.5	Release 4.0 - December 1998	27
15.6	Release 5.0 - June 2000	27
15.7	Release 6.0 - November 2001	28
15.8	Release 6.1 - June 2002	28
15.9	Release 6.3 – December 2002	30
15.10	Release 7.0 – July 2003	30
15.11	Release 7.2 – December 2004	30
15.12	Release 7.3 – February 2006	31
15.13	Release 7.31 – April 2006	32
15.14	Release 7.4 – January 2007	32
15.15	Release 7.5 – May 2007	33
16	Operating system compatibility	34
16.1	VAX/VMS	34
16.2	SPARC/Solaris	34
16.3	Windows NT, 2000 and XP	34
16.4	Intel/Linux	34
16.5	Apple PowerPC/OS X	34
	Document Control and References	35
	Changes history	35
	Changes forecast	35
	Document references	35
	File under	35

1

1 Introduction

Continued development of the SPARK Toolset has resulted in the development of a number of useful features that have been incorporated into Toolset Release 7.5.

This document describes changes in the behaviour of all variants of the SPARK Toolset Release 7.5 compared to Release 7.4.

1

Tool Development and Support
SPARK Toolset Release Note–Release 7.5

EXM/RN
Issue: 1.1

2 Contact Information

For further information about this document please contact Praxis High Integrity Systems:

By phone: +44 (0)1225 823829 (direct line), +44 (0)1225 466991 (exchange)

By FAX: +44 (0)1225 469006

By email: sparkinfo@praxis-his.com

1

Tool Development and Support
SPARK Toolset Release Note–Release 7.5

EXM/RN
Issue: 1.1

3 SPARK language changes

None.

1

4 SEPR 2124 – Important Information for Users

4.1 Description

SEPR 2124 concerns a case where the Examiner fails to generate an RTC VC where the canonical semantics of Ada require a check for `Constraint_Error` to be performed. This means that all the remaining VCs for a subprogram can all be reported as “True” by the Simplifier, even if a potential run-time error does exist.

The exact circumstances are as follows:

- A procedure exists with a formal parameter which is of a scalar type of mode “in out” or mode “out”, and
- Another program unit calls that procedure with an actual parameter which is a restricted subtype of the type of the formal parameter, and
- The actual parameter is a field of a record object or an element of an array object.

4.2 SPARK Examiner versions affected

All Examiner versions from 4.0 through 7.4 inclusive are affected, although the use of array elements as actual parameters only became legal in SPARK in release 7.1.

4.3 Test case

The code below illustrates the problem for both a record field and an array element:

```
package P is
  type Colour is (Red, Green, Blue, Purple);
  subtype Nice_Col is Colour range Green .. Purple;

  type R is record
    F1 : Natural;
    F2 : Nice_Col;
  end record;

  type I is range 1 .. 10;
  type A is array (I) of Nice_Col;
  -- Body of this not important, but note that it returns
  -- a Colour, not a Nice_Col
```

1

```
    procedure F (Q : in Natural; C : out Colour);
    --# derives C from Q;

end P;

with P;
--# inherit P;
package Test is
    procedure Op1 (X : in Natural;
                  Y : out P.Colour);
    --# derives Y from X;

    procedure Op2 (X : in Natural;
                  Y : out P.Colour);
    --# derives Y from X;

end Test;

package body Test is

    -- record field case
    procedure Op1
        (X : in Natural;
         Y : out P.Colour)
    is
        W : P.R;
    begin
        -- The following procedure call should generate a VC to
        -- show that (W.F2 in P.Colour) -> (W.F2 in P.Nice_Col)
        -- which is unprovable.
        -- Examiners 4.0 through 7.4 do NOT generate this VC.
        P.F (X, W.F2);
        Y := W.F2;
    end Op1;

    -- array element case
    procedure Op2
        (X : in Natural;
         Y : out P.Colour)
    is
        W : P.A;
    begin
        W := P.A'(others => P.Green);
    end Op2;
end Test;
```

1

```
-- The following procedure call should generate a VC to
-- show that (W (1) in P.Colour) -> (W (1) in P.Nice_Col)
-- which is unprovable.
-- Examiners 4.0 through 7.4 do NOT generate this VC.
P.F (X, W (1));
Y := W (1);
end Op2;

end Test;
```

4.4 Workaround

The problem does not occur if a “whole variable” is passed as an actual parameter – the correct VC is generated in this case.

Therefore, code which always passes “whole variables” as parameters cannot be affected by this problem. One possible work-around is to identify code that may be at risk, and to re-introduce the use of whole variables for all actual parameters.

Code at risk of being affected by this problem can be identified using the procedure described in section 4.5 below.

4.5 Recommended action

A full release of the toolset at version 7.5 is now available. This Examiner correctly generates the VC that was previously missing and generates a new warning to notify users of code that may be at risk. The new warning is issued by the VC Generator, so the analysis must be run with the VC Generator enabled

For example, when generating VCs for procedure Test.Op1 in the test case above, Examiner 7.5 reports:

```
45          P.F (X, W.F2);

--- Warning:420: Instance of SEPR 2124 found. An extra VC will be
generated here and must be discharged to ensure absence of
run-time errors. Please contact Praxis High Integrity Systems
for assistance with this issue.
```

We recommend a complete “back to back” analysis of an entire project. In short:

- Examine entire project using Examiner release 7.4 (or whatever version is currently in use), generating VCs using the “/exp” (Windows) or “-exp” (Solaris) option, plus the “plain” option.

1

- Re-Examine the entire project using Examiner 7.5 using identical options.
- Perform a mechanical “diff” of the listing and report files produced by 7.4 against those produced by 7.5.

If warning 420 (as above) appears in any of the output generated by Examiner 7.5, then those VCs should be re-simplified and re-checked for potential run-time errors.

1

5 SEPR 1979 – Important Information for Users

5.1 Description

SEPR 1979 concerned the generation of VCs for the right-hand side of a modular exponentiation operator, and was implemented in release 7.4 of the Examiner. This SEPR, though, also causes an additional change in the Examiner's behaviour that all users should be aware of.

This SEPR now causes the Examiner to generate a run-time check VC for an explicit conversion of a numeric expression onto a modular subtype when the conversion appears as part of a larger compound expression. This VC was not generated by Examiner releases 7.31 and earlier.

5.2 SPARK Examiner versions affected

Examiner versions 6.1 through 7.31 are affected by this problem.

Examiner versions 7.4 and later are NOT affected.

5.3 Test case

Given the following declarations:

```
type T is mod 256;  
subtype S is T range 0 .. 7;  
X : T;  
C : Integer;
```

And the statement:

```
X := X + S (C);
```

Examiner releases 7.31 and earlier do NOT generate conclusions in the VC to show that "C in S" for the modular subtype conversion.

5.4 Workaround

Avoid modular subtype conversions in compound expressions.

1

5.5 Recommended action

If you are using Examiner release 7.31 or earlier and modular subtypes, we recommend a complete “back to back” analysis. In short:

- Examine entire project using Examiner release 7.31 (or whatever version is currently in use), generating VCs using the “/exp” (Windows) or “-exp” (Solaris) option, plus the “plain” option.
- Re-Examine the entire project using Examiner 7.5 with identical options.
- Perform a mechanical “diff” of the listing and report files produced by 7.31 against those produced by 7.5.

If additional conclusions and/or VCs are generated by release 7.5, then those VCs should be re-simplified and re-checked for potential run-time errors.

1

6 Examiner changes

This section documents other significant changes to the SPARK Examiner made for releases 7.4d01 to 7.5. Where appropriate, SPARK Examiner Performance Report (SEPR) numbers are given.

6.1 Increase in per-file justification limit (SEPR 2112)

The number of justifications allowed per file has been increased from 100 to 300.

6.2 New RTE checks for Ada.Real_Time (SEPR 2107)

The Examiner now generates an RTE check for all instances related to `Ada.Real_Time.Time` and `Ada.Real_Time.Time_Span` where previously a warning about the lack of an RTE check would have been emitted. This change only applies when the Ravenscar profile is selected.

6.3 Hypotheses for components unchanged by a call (SEPR 2108)

The Examiner now generates hypotheses for unchanged components (record fields and array elements) when a component of a composite is of a private type and is an actual parameter that is exported from the called procedure.

6.4 Proof functions and abstract own variables (SEPR 2109)

A proof function that refers to abstract state now obtains the correct view (i.e. pre-versus-post refinement view). Previously, declaration within a private part could be overridden if a body was also presented to the Examiner, causing the post-refinement view to be used incorrectly.

6.5 Failure to initialize a protected object is a flow error (SEPR 2113)

The failure to initialize a protected component at the point of declaration has been changed from a semantic error to a flow error. This allows the error to be justified using an `accept` annotation.

6.6 Visibility of embedded package own variables from a for loop (SEPRs 1696, 2067)

A correction has been made so that the own variable of an embedded package is visible from within a “for” loop in the sequence of statements that follows that embedded package.

1

6.7 Space allowed in Windows command line (SEPR 2028)

A space character is now permitted to separate filenames on the Examiner command-line on Windows platforms. This allows for the use of file globs such as <subsystem>*.adb

6.8 Accept annotation following plain loop (SEPR 2075)

The Examiner now allows an accept annotation to follow a plain loop in the main subprogram or a task body, even if that unit has a hidden exception handler part.

6.9 Size clause value must be static (SEPR 2093)

The Examiner now implements the Ada rule that the value given in a Size representation clause must be static.

6.10 Number of arguments for pragmas (SEPR 2096)

While the Examiner cannot check the correctness of pragma arguments in general, it now checks that the number of arguments is at least correct for all pragmas defined by the Ada LRM.

6.11 Examiner memory usage (SEPR 2123)

Examiner memory usage has been substantially reduced between 7.4 and 7.5.

6.12 New warning control keywords: default_loop_assertions and real_rtcs (SEPR 2126)

It is now possible to control the reporting of Semantic Warnings 402 and 405 using the default_loop_assertions and real_rtcs keywords respectively in the warning control file.

1

7 SPADE Simplifier

Toolset release 7.5 ships with Simplifier version 2.32. This includes the following improvements:

7.1 Improved reasoning for inequalities involving transitivity (SEPR 2120)

The Simplifier has improved capability to deduce conclusions where a proof by transitivity over the integers is required. This improves simplification of the VCs that arise from SPARK where a loop is used to iterate over an array data structure.

7.2 Enhanced reporting of user defined rules (SEPR 2106)

The simplifier log (SLG) file now includes the following summaries relating to user defined rules:

- The user rule files read in by the simplifier,
- Any syntax errors found in the user rule files,
- For each VC the user defined rules used in proving a conclusion or promoting a fact to a hypothesis, and
- An overall summary defining the user defined rules which have been used in proving part or all of a VC.

As well as providing useful information when creating and developing user defined rules, the summaries are used by POGS for better reporting of their use, see section 9.1 .

7.3 SPARKSimp

No change between 7.4 and 7.5.

1

Tool Development and Support
SPARK Toolset Release Note–Release 7.5

EXM/RN
Issue: 1.1

8 SPADE Proof Checker

No change between 7.4 and 7.5.

1

9 POGS

Toolset release 7.5 ships with POGS version 4.7 and includes the following improvement:

9.1 Description of user-defined proof rules and summary-only option (SEPR 2106)

A new 's' switch can be used to cause POGS to output only the final summary for cases in large projects where the '.sum' file would otherwise be very big. Extra information is provided on the use of user-defined proof rules in all cases, with information on use of rules given on a per-VC and per-conclusion basis. The final summary is also expanded to indicate what proportion of VCs discharged by the Simplifier were aided by user-defined proof rules.

1

10 SPARKFormat

Toolset release 7.5 ships with SPARKFormat version 7.5 and includes the following improvement:

10.1 Empty line in global annotation should not signal end of annotation (SEPR 2125)

A rare issue is corrected where an empty line in the middle of a global annotation would be interpreted as the end of the annotation (a problem in some auto-generated code).

1

Tool Development and Support
SPARK Toolset Release Note–Release 7.5

EXM/RN
Issue: 1.1

11 SPARKMake

No change between 7.4 and 7.5.

1

12 Backward incompatibilities

There are no known backward incompatibilities introduced between Examiner Releases 7.4 and 7.5.

1

13 User manual change summary

The following user manuals have been changed between Examiner Releases 7.4 and 7.5.

- SPARK Examiner User Manual
- Simplifier User Manual
- POGS User Manual

13.1 Examiner User Manual

- Documentation of additional warning control keywords.
- Documentation of new warnings.

13.2 Simplifier User Manual

- Documentation of output of additional information on use of user-defined proof rules.

13.3 POGS User Manual

- Documentation of new 's' switch for summary-only output.
- Documentation of output of additional information on use of user-defined proof rules.

14 Limitations and known errors

14.1 Tool limitations

This section describes limitations of the Examiner tool arising mainly from incomplete implementation of planned features. Where appropriate a SPARK Examiner Performance Report (SEPR) number is given.

14.1.1 General

- 1 The SPARK 95 language definition removes the distinction between initial and later declarative items; this distinction remains in force in the Examiner that requires SPARK 83 declaration orders even in SPARK 95 mode. (SEPR 813)
- 2 The Examiner does not yet permit the use of 8-bit characters in SPARK 95 userdefined identifiers. (SEPR 818)
- 3 Universal expressions in a modular context may sometimes require type qualification. (SEPR 1591)
- 4 The Examiner does not yet permit the use of “use type” following an embedded package specification. (SEPR 747)
- 5 The Examiner does not yet permit the renaming of packages in the same way that subprograms can be renamed. (SEPR 1391)
- 6 The Examiner does not yet allow the 'Base attribute when not used as a prefix. (SEPR 1114)
- 7 The Examiner does not yet allow S'Range where S is scalar. (SEPR 1115)

14.1.2 Verification Condition Generation and Run-time Checks

- 1 Ada string inequality is not modelled. (SEPR 712)
- 2 VCs involving string catenation that includes the character ASCII.NUL will be incorrect. (SEPR 661)
- 3 Aggregates of multi-dimensional arrays cannot be modelled although aggregates of arrays of arrays can. (SEPR 590)
- 4 Verification conditions involving real numbers are evaluated using infinite precision or perfect arithmetic; this allows the correctness of an algorithm to be shown but cannot guard against the effects of cumulative rounding errors for example.
- 5 The Examiner does not generate VCs for package initialization parts. Statically determinable constraint errors will be detected during well-formation checking of package initialization. (SEPR 288)

1

- 6 The VC Generator cannot model the implementation-dependent attributes of floating and fixed-point types; see section 14.1.3.

14.1.3 Attribute limitations

14.1.3.1 Unimplemented attributes

The following attributes are officially supported by SPARK according to the language definition, but are not yet implemented by the Examiner. The Examiner will generate error number 30 (“Attribute XXX is not yet implemented in the Examiner”) if you try to use them.

- Adjacent
- Compose
- Copy_Sign
- Leading_Part
- Remainder
- Scaling
- Exponent
- Fraction
- Machine
- Model
- Rounding
- Truncation
- Unbiased_Rounding

Note that these are all function-like attributes concerning floating- and fixed-point types.

14.1.3.2 Unevaluable attributes

The Machine_* and Model_* attributes are accepted by the Examiner, but it does not know how to statically evaluate them since they are inherently implementation dependent. For example, the package:

1

```
package F is
  type T is digits 6 range -10.0 .. 10.0;
  C : constant := T'Machine_Emax;
end F;
```

is legal SPARK, but the Examiner does not know the actual numeric value of C.

14.2 Known error summary

This section lists known errors in the Examiner that are awaiting investigation and correction.

- 1 The SPARK rule that array actual parameters must have the same bounds as the formal parameter is not checked for function parameters where the actual parameter is a subtype of an unconstrained array type. Since subtype bounds are static in SPARK errors of this kind should be detected by an Ada compiler. If not an unconditional run-time error will occur. (SEPR 1060)
- 2 The Examiner permits the body of a subprogram to be entirely made up of proof statements thus breaching the Ada rule that at least one Ada statement must be present. (SEPR 278)
- 3 Where a package declares two or more private types the Examiner permits mutual recursion between their definitions in the private part of the package. (SEPR 848)
- 4 The Examiner does not take due account of a range constraint when determining the subtype of a loop variable; this affects completeness checking of case statements within the loop. For example **for I in Integer range 1..4 loop** would require only values 1, 2, 3 and 4 to be covered by the case statement. (SEPR 693)
- 5 When summarising the counts of pragmas found during an analysis the totals may depend on whether units are selected via the command line (or metafile) or using the index mechanism. The difference affects only pragmas placed *between* program units and arises because placing a file name on the command line causes the entire *file* to be analysed whereas selecting it using indexes causes only the required *unit* to be read from the file. (SEPR 483)
- 6 Justifications involving a subcomponent of a record can cause data flow analysis to become overly pessimistic when considering expressions involving that record within the same subprogram. The result is spurious Flow Errors, and as a result justifications should not be used in this instance. (SEPR 2132)

1

15 Change Summary from Release 2.0

A release note detailing changes from the previous version accompanies each Examiner Release; this section simply summarises the various changes that have been made.

15.1 Release 2.0 - November 1995

Release 2.0 added:

- static expression evaluation;
- variable initialization at declaration;
- full-range scalar subtypes; and
- operator renaming in package specifications.

15.2 Release 2.1 - July 1996

Release 2.1 added:

- facilities for proof of absence of run-time errors

15.3 Release 2.5 - March 1997

Release 2.5 was distributed with “High Integrity Ada - the SPARK Approach” and provided initial facilities for SPARK 95

15.4 Release 3.0 - September 1997

Windows NT was supported for the first time with this release. Release 3.0 also added:

- additional SPARK 95 support;
- flow analysis of record fields;
- command line meta files;
- named numbers;
- unqualified string literals;
- moded global annotations; and

1

- optional information flow analysis.

15.5 Release 4.0 - December 1998

With Release 4.0 we upgraded all users to a single product standard supporting SPARK 83, SPARK 95 and analysis options up to an including proof facilities. New features were:

- full implementation of public and private child packages;
- default switch file; and
- provision of the INFORMED design document.

15.6 Release 5.0 - June 2000

- Enhanced proof support:
 - I. facilities for proof of programs containing “abstract state”;
 - II. addition of quantified expressions;
 - III. proof rule generation for enumeration types;
 - IV. identification of the kind and source of each VC;
 - V. suppression of trivially true VCs;
 - VI. Proof Obligation Summariser tool (POGS)
- Optional HTML output files with hyperlinks that can be “browsed” interactively
- Better support for common Ada file naming conventions
- User-selectable annotation character
- Improved suppression of analysis were results might otherwise be misleading
- Static expression evaluation in proof contexts
- Singleton enumeration types
- Revised SPARK_IO package
- Error numbering

1

15.7 Release 6.0 - November 2001

- Introduction of “external variables” to simplify modelling of the interactions between a SPARK program and its external environment.
- Addition of the “null derives” annotation to describe information flows which affect only the external environment.
- Introduction of modular types
- Use of loop labels in exit statements
- Use of global modes on function subprograms
- Extended support for predefined types such as Long_Integer
- Simplified run-time check generation for own variables
- Relaxation of need for mandatory type announcement of own variables
- Plain output option to simplify comparisons of Examiner output files
- Platform-independent switch files and metafiles
- Support for intentionally infinite loops
- Detection of own variables that can never be initialized
- Detection of unusable private types
- Extra refinement checks on global variables when performing data flow analysis
- Detection of unnecessary others clause in case statements
- Extensions to the POGS tool
- Improved error messages to distinguish different cases of variables which are “not declared or visible”
- Improved SPADE Simplifier Release 2.0
- New “SPARKSimp” Tool

15.8 Release 6.1 - June 2002

- Introduction of tagged types

1

- Introduction of type assertion annotations
- Introduction of modular subtypes
- Introduction of the configuration file
- Introduction of the `help` command line switch
- Demo Examiner now runs on Linux.
- VCG generation for inherited operations of tagged types
- Improved handling of null derives
- Attributes 'Floor and 'Ceiling implemented
- Detection of duplicate record fields
- Improved overflow checks on universal integer expressions
- Corrected handling of loop invariants in while loops
- Strengthened behaviour of `/noecho` option
- Trapping non-positive accuracy in real type declaration
- Recursion in meta-files and index-files
- Improved handling of Address clauses
- Improved handling of Import and Interface pragmas
- VCG Modelling of Boolean membership operators
- Simplification of common Integer inequalities
- Simplification of common enumerated inequalities
- Simplification of VCs involving quantified expressions
- Simplifier performance
- Checker has new built-in rule families: MODULAR, BITWISE and ENUMERATION
- Proved by review option in POGS

1

15.9 Release 6.3 – December 2002

Release 6.3 of the toolset was constructed to accompany the textbook “High Integrity Software: The SPARK Approach to Safety and Security” by John Barnes, but was not delivered to other users. Its main new features were:

- Slight revision to the rules regarding the placement of tagged type declarations.
- Correction to the modelling of Boolean type membership operators in the verification conditions.
- Support for generating VCs that allow the verification of the Liskov Substitution Principal (LSP) for tagged types and their operations.
- Dramatically improved performance of the Simplifier, particularly in the simplification of quantified expressions.

15.10 Release 7.0 – July 2003

Release 7.0 of the toolset comprised:

Examiner version 7.0

Simplifier version 2.12

Release 7.0 added:

- Ravenscar profile extensions to the language.
- Support for Ada.Interrupts and Ada.Real_Time in the configuration file.
- The new /noduration command line switch.
- VC generation for unconstrained formal parameters.
- Suppression of VC generation for illegal function bodies.
- New “SPARKFormat” tool.

15.11 Release 7.2 – December 2004

Release 7.2 of the toolset comprised:

Examiner version 7.2

Simplifier version 2.17

1

Release 7.2 added:

- Unconstrained string constants to the language.
- Instantiation of Unchecked_Conversion to the language.
- (Full) record subtypes to the language.
- Declaration of subprograms in the private part of packages.
- Refined proof annotations for private types.
- % suffix for referring to value of variable on entry to loop in proof contexts.
- Extra hypotheses for local variables.
- Suppression of VC generation for illegal function bodies.
- Replacement rules for composite constants.
- Concurrent simplification with SPARKSimp.
- Improved simplification of VCs with large structured objects.
- Improved simplification of arithmetic and logical expressions.
- New “SPARKMake” tool.

15.12 Release 7.3 – February 2006

Release 7.3 of the toolset comprised:

Examiner version 7.3

Simplifier 2.22

Checker 2.06

Significant features of this release included:

- Improved VC Generation.
- New “error explanations” switch for Examiner.
- Generation of proof rules for ‘Size.
- Improved diagnosis and reporting of common syntax errors.

1

- Error and warning count summary in Examiner output.
- Use of pragma Import to complete an external own variable.
- Correction to FDL declaration order for private and announced types.
- New Simplifier tactics for dealing with rational inequalities and Examiner-generated proof rules.
- User-defined proof rules for the Simplifier.
- New Checker rules for MODULAR expressions.
- Significant performance improvement for Simplifier and Checker.

15.13 Release 7.31 – April 2006

Release 7.31 of the toolset comprised:

Examiner version 7.31

Simplifier 2.24

Checker 2.07

Significant features of this release included:

- Correction to flow analysis of array element “out” parameters.
- Port of “Demo” toolset to Mac OS X.
- New /typecheck option in Simplifier.
- Improved validity checking of user-defined proof rules in the Simplifier.
- Correction to ARITH proof rules in the Checker.

15.14 Release 7.4 – January 2007

Release 7.4 of the toolset comprised:

Examiner version 7.4

Simplifier 2.30

Checker 2.08

1

Significant features of this release included:

- The 'accept' annotation.
- Conditional flow errors are now reported as errors.
- The Simplifier supports user defined proof rules.
- SPARKFormat supports a wider range of annotations.
- SPARKMake has the ability to generate relative pathnames.

15.15 Release 7.5 – May 2007

Release 7.5 of the toolset comprised:

Examiner version 7.5

Simplifier 2.32

Checker 2.08

Significant features of this release included:

- Correct VC generation for record fields and array elements used as "in" or "in out" parameters.

1

16 Operating system compatibility

16.1 VAX/VMS

The toolset is not supported on VAX/VMS.

16.2 SPARC/Solaris

The toolset is compatible with Solaris 5.6 through to 10 including those with a 64-bit kernel.

16.3 Windows NT, 2000 and XP

The toolset is compatible with Windows NT 4.0, Windows 2000, and Windows XP. The executables are also known to work on Windows 95 and 98; however, use of the toolset on these operating systems is unsupported. The FlexLM licence manager software only runs on Windows NT, Windows 2000, or Windows XP.

16.4 Intel/Linux

All the toolset, with the exception of the Checker, is compatible with Intel-based Linux operating systems. Only the "FreeDemo" version of the toolset is currently available for Linux to support buyers of John Barnes' "SPARK Book." If you require a full professional SPARK toolset for Linux, then please contact us.

16.5 Apple PowerPC/OS X

All the toolset, with the exception of the Checker, is compatible with Apple's PowerPC OS X/Darwin operating systems. Only the "FreeDemo" version of the toolset is currently available for OS X to support buyers of John Barnes' "SPARK Book." If you require a full professional SPARK toolset for OS X, then please contact us.

1

Document Control and References

Praxis High Integrity Systems Limited, 20 Manvers Street, Bath BA1 1PX, UK.
Copyright © Praxis High Integrity Systems Limited 2007. All rights reserved.

Changes history

Issue 0.1 (17th January 2007):	First draft for release 7.4d01
Issue 0.2 (14th March 2007):	Draft for release 7.4d02
Issue 0.3 (23rd March 2007)	Release 7.4d02
Issue 0.4 (5th April 2007)	Release 7.4d03
Issue 0.5 (23rd April 2007)	Release 7.4d04
Issue 0.6 (26th April 2007)	Include full description of SEPR 2124 for 7.4d04
Issue 0.7 (30th April 2007)	Release 7.5
Issue 0.8 (16 May 2007)	Include updates to simplifier for release 7.5
Issue 1.0 (25 May 2007)	Bump up to Issue 1.0 for release 7.5 following review
Issue 1.1 (29 May 2007)	Add section on impact of SEPR 1979

Changes forecast

None

Document references

File under

SVN: trunk/userdocs/Examiner_RN_7p5.doc